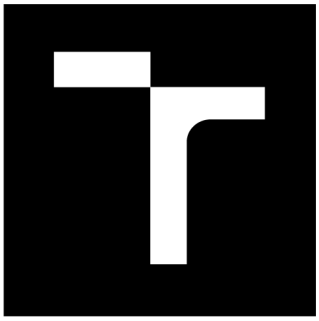
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2017 Pavol Lieskovský



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

STROJOVÉ POROZUMĚNÍ TEXTOVÝM ZPRÁVÁM POUŽÍVANÝCH V LETECTVÍ
MACHINE UNDERSTANDING FOR TEXT MESSAGES USED IN AVIATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Pavol Lieskovský
AUTHOR

VEDOUCÍ PRÁCE — Ing. Lukáš Povoda
SUPERVISOR

BRNO 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Teleinformatika**
Ústav telekomunikací

Student: Pavol Lieskovský ***ID:*** 173689
Ročník: 3 ***Akademický rok:*** 2016/17

NÁZEV TÉMATU:

Strojové porozumění textovým zprávám používaných v letectví

POKYNY PRO VYPRACOVÁNÍ:

Vytvořte algoritmus, který z textových zpráv používaných v avionice (tzv. NOTAM) získá důležité informace, které jsou ve zprávě obsaženy. Navrhněte pravidla pro získání informace o tvaru složených oblastí a implementujte je v jazyce Java. Pro účely prezentace narhněte jednoduché rozhraní s mapou, do které vykreslete detekované oblasti. Program rozšiřte o detekci kolize oblasti s letovým plánem.

DOPORUČENÁ LITERATURA:

[1] POVODA, L.; BURGET, R.; MAŠEK, J.; CVRK, L. Automatické rozpoznávanie emócií z českého textu pomocou umelej inteligencie. Elektrorevue - Internetový časopis (http://www.elektrorevue.cz), 2015, roč. 17, č. 1, s. 15-18. ISSN: 1213-1539.

[2] CHIEU, Hai Leong; NG, Hwee Tou. Named entity recognition: a maximum entropy approach using global information. In: Proceedings of the 19th international conference on Computational linguistics-Volume 1. Association for Computational Linguistics, 2002. p. 1-7.

Termín zadání: 1.2.2017 ***Termín odevzdání:*** 8.6.2017

Vedoucí práce: Ing. Lukáš Povoda
Konzultant:

doc. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto práca sa zaoberá problematikou textovej správy NOTAM, ktorá sa používa v letectve. Dokumentuje rozdiel medzi textovou a digitálnou podobou správy NOTAM, špeciálne typy správ NOTAM a položky z ktorých sa správa NOTAM skladá. Popisuje syntax a funkcie programu, ktorý bol v rámci práce vytvorený. Program je plne schopný správneho spracovania a naparsovania správy NOTAM. Program dokáže do mapy vykresliť oblasti jednotlivých správ NOTAM a tiež poskytuje funkciu detekcie kolízie týchto oblastí s letovým plánom.

KĽÚČOVÉ SLOVÁ

NOTAM, parsovanie, vykreslenie oblasti, detekcia kolízie

ABSTRACT

This work deals with problems of NOTAM in text format, which is used in aeronautics. It documents the difference between text and digital format of NOTAM, special types of NOTAM messages and items from which the NOTAM consist of. It describes syntax and the functions of program, which was made within the frame of this thesis. The program is fully capable of correct parsing and processing of the NOTAM. The program can display each area of processed NOTAM messages in map and also provides detection of collision between these areas and flight plan.

KEYWORDS

NOTAM, parsing, area delineation, collision detection

LIESKOVSKÝ, Pavol *Strojové porozumění textovým zprávám používaných v letectví*: bakalárska práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2016. 55 s. Vedúci práce bol Ing. Lukáš Povoda

VYHLÁSENIE

Vyhlasujem, že som svoju bakalársku prácu na tému „Strojové porozumění textovým zprávám používaných v letectví“ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Lukášovi Povodovi za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

Brno . . . . . .

. . . . . .

podpis autora(-ky)

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

Úvod **12**

1 Rozbor problematiky **14**

1.1 Typy správ NOTAM 15

1.1.1 NOTAM vs D–NOTAM 15

1.1.2 Špeciálne typy 16

1.2 Skladba správy NOTAM 18

1.2.1 Séria a identifikátor 18

1.2.2 Položka Q) 18

1.2.3 Položka A) 22

1.2.4 Položka B) 22

1.2.5 Položka C) 23

1.2.6 Položka D) 23

1.2.7 Položka E) 23

1.2.8 Položka F) a G) 24

1.2.9 Dekódovanie správy NOTAM 24

2 Návrh riešenia **26**

2.1 Trieda TextReader 26

2.1.1 Metóda `loadNotams()` 27

2.2 Trieda Parser 28

2.2.1 Metóda `addNewNotam(String _key, String _value)` 29

2.2.2 Metóda `notamParser()` 29

2.2.3 Metóda `getAdaptedParsedNotams()` 31

2.3 Triedy reprezentujúce objekty 33

2.3.1 Trieda Notam 33

2.3.2 Trieda FlightPlan 34

2.4 Trieda BorderComputer 35

2.4.1 Metóda `computeBorderCoordinates( ... )` 36

2.4.2 Metóda `computePoints( ... )` 37

2.5 Trieda AreaComputer 38

2.5.1 Metóda `computeAreaIntersection( ... )` 39

2.5.2 Metóda `computeFlightCorridor(List<Double> flightPath)` 40

2.5.3 Metóda `computeNotams(HashMap <String, List <String>> parsedNotams)` 41

2.6 Trieda Gui 42

2.6.1 Panel rightTopPanel 43
2.6.2 Panel rightMidPanel 45
2.6.3 Panel rightbottomPanel 47

3 Záver **49**

Literatúra **50**

Zoznam symbolov, veličín a skratiek **52**

Zoznam príloh **53**

A Obsah priloženého CD **54**

ZOZNAM OBRÁZKOV

1.1 Grafická vizualizácia D-NOTAMu[5] 15
1.2 Spôsob zápisu série a identifikátoru 18
1.3 Skladba položky Q) 19
1.4 Ukážka zápisu FIR v položke Q) 20
1.5 Rozdelenie strednej Európy podľa FIR[6] 20
1.6 Ukážková správa NOTAM 24
2.1 Ukážková naparsovaná správa NOTAM 31
2.2 Ukážková naformátovaná správa NOTAM 33
2.3 Vykreslenie letového plánu definovaného dvoma súradnicami 40
2.4 Grafické rozhranie programu 42
2.5 Rozloženie komponentov v panely `rightTopPanel` uživateľského rozhrania 43
2.6 Rozloženie komponentov v panely `rightMidPanel` uživateľského rozhrania 45
2.7 Rozloženie komponentov v panely `rightBottomPanel` uživateľského rozhrania 47
A.1 Adresárová štruktúra priloženého CD 54

ZOZNAM TABULIEK

1.1 Označenia intenzity pohybu vtáctva v správe BIRDTAM 17
1.2 Možnosti a význam označenia v tretom poli položky Q) 21
1.3 Možnosti a význam označenia vo štvrtom poli položky Q) 21
1.4 Možnosti a význam označenia piatého pola položky Q) 22

ZOZNAM VÝPISOV

2.1 Syntax triedy `TextReader` 26
2.2 Spracovanie položky E) 27
2.3 Syntax triedy `Parser` bez obsahu tiel metód 28
2.4 Rozdelenie a odstránenie prebytočných znakov položky E) 30
2.5 Podmienky spracovania slova OF 31
2.6 Syntax triedy Notam 33
2.7 Syntax triedy `FlightPlan` 34
2.8 Syntax konštruktoru triedy `BorderComputer` 36
2.9 Syntax zvolenia správneho for cyklu pre pridanie súradníc popisujúcich hranicu 38
2.10 Syntax funkcie volanej tlačidlom draw NOTAMs 44
2.11 Syntax funkcie volanej tlačidlom delete selected point 47
2.12 Syntax funkcie volanej tlačidlom zoom in 48

ÚVOD

Dopyt po preprave leteckou dopravou je čoraz väčší, čoho dôsledkom je aj jej rastúca hustota. Na londýnskom letisku Heathrow, ktoré patrí medzi najväčšie letiská sveta, sa denne uskutoční v priemere až 1300 letov čo znamená, že lietadlo priletí alebo odletí približne každú jednu minútu. S rastom hustoty letov priamo úmerne rastie aj náročnosť a komplikovanosť riadenia letovej prevádzky, ktoré je zodpovedné za organizovanie letov a ich bezpečnosť. Aby bola hustota leteckej dopravy bezpečnostne zvládnuteľná, boli zavedené medzinárodné systémy a pravidlá, vďaka ktorým je možná prehľadná kontrola vzdušného priestoru a bezpečné vykonanie všetkých letov.[11]

Jeden z kľúčových prvkov v letectve zodpovedný za bezpečné prevedenie letu je správa Notice To Airmen (NOTAM). Predmetom tejto práce je práve správa NOTAM–konkrétne jej staršia, textová, neštrukturovaná podoba, ktorá nie je strojovo spracovateľná. Pilot v rámci predletovej prípravy dostane množstvo papierov so správami NOTAM, ktorých spracovanie a dekódovanie je práve jeho úlohou. Často sa stáva, že pilot nemá dostatok času a spracovanie všetkých správ nie je možné. Z tohto dôvodu bola zavedená plne štrukturovaná podoba NOTAM správ určená primárne na strojové spracovanie. V prevádzke je zavedená už niekoľko rokov, avšak len v určitých krajinách. Vo východných krajinách ako sú napríklad Pakistan, Uzbekistan alebo aj svetová veľmoc Rusko sa stále využíva neštrukturovaná podoba.[3]

Myšlienkou tejto práce je vytvoriť program, ktorý textovú správu NOTAM správne strojovo spracuje, dokáže vykresliť oblasť jej platnosti do mapy a taktiež dokáže detekovať kolíziu oblasti s letovým plánom. Zameranie je hlavne na správy popisujúce zložitú oblasť a jej správne spracovanie do štrukturovanej podoby vhodnej na grafické zobrazenie. Program schopný takéhoto spracovania predstavuje pre leteckú dopravu značné zjednodušenie a zefektívnenie. Povinnosť pilota spracovať textové správy NOTAM preberá software, ktorý ich pilotovi zobrazí v grafickej podobe. Takéto zobrazenie ušetrí množstvo času a je efektívnejšie, plus sa zamedzí strate informácií potrebných pre bezpečné prevedenie letu, spôsobenej nedostatkom času na dekódovanie všetkých správ obsiahnutých v predletovej príprave.

Práca je štrukturovaná nasledovne: Kapitola 1 sa zaoberá teoretickou časťou práce. Začiatok kapitoly je venovaný zoznámeniu s pojmom správa NOTAM. V podkapitole 1.1 sú bližšie rozobrané a popísané typy správy NOTAM. Nachádza sa tu porovnanie štrukturovanej a neštrukturovanej podoby správy a popis špeciálnych typov ASHTAM, BIRDTAM, SNOWTAM. Podkapitola 1.2 obsahuje popis skladby správy NOTAM. Sú v nej postupne rozobraté všetky položky z ktorých sa správa skladá, vrátane príkladov. Na konci je ukážkové dekódovanie celej správy NOTAM. Kapitola 2 popisuje praktickú časť práce–program určený na spracovanie a vykres-

lenie správ NOTAM. Kapitola je členená na šesť podkapitol, pričom každá podkapitola sa venuje jednej konkrétnej triede. Výnimkou je podkapitola 2.3, ktorá pokrýva dve triedy. V rámci každej podkapitoly je bližšie vysvetlený syntax a logický význam triedy, ktorou sa podkapitola zaoberá. Podkapitoly sú následne delené na odstavce, ktoré sú venované konkrétnym metódam triedy s výnimkou podkapitoly 2.6, kde sú jednotlivé odstavce delené podľa rozloženia komponentov grafického užívateľského rozhrania. Pre lepšie pochopenie textu sa v podkapitolách nachádzajú výpisky zobrazujúce časti kódu tried.

1 ROZBOR PROBLEMATIKY

Notice To Airmen (NOTAM) v doslovnom preklade oznámenie letcovi je správa, ktorá nachádza svoje uplatnenie v leteckej doprave. Pre správne a najmä bezpečné fungovanie letovej prevádzky sú správy NOTAM kľúčové. Tieto správy sú určené najmä pre pilotov, letových dispečerov ale aj ostatní personál podieľajúci sa na letovej prevádzke. Obsahujú informácie o rôznych krátkodobých alebo náhlych zmenách, ktoré predstavujú pre letovú prevádzku nebezpečenstvo, pričom ich trvanie môže byť obmedzené na niekoľko dní, hodín, týždňov alebo až do ďalšej zmeny. NOTAM správy sú vytvárané a distribuované štátnym leteckým úradom štátu, v ktorom je správa vydaná a ich distribúcia prebieha prostredníctvom siete ako je napr. Aeronautical Fixed Telecommunication Network (AFTN). Podľa štatistických údajov uvedených v dokumente [4] ktorý je vydaný pod International Civil Aviation Organization (ICAO) počet správ NOTAM každým rokom celosvetovo stúpa v priemere o 10% a denne je platných v priemere 18000 NOTAM správ po celom svete. Z tohto dôvodu sa stále pracuje na vývoji a zefektívnení správ NOTAM a práce s nimi.[16][4] Niektoré z dôvodov na vydanie správy NOTAM:

- uzavrenie alebo významné zmeny v prevádzke letiska/heliportu alebo dráh,
- prerušenie prevádzky alebo opätovné uvedenie do prevádzky hlavných častí letiskových svetelných systémov,
- zmeny alebo obmedzenie výdaju paliva, oleja alebo kyslíka, ktoré sú k dispozícii,
- výskyt nebezpečenstva ktoré môže mať vplyv na letovú prevádzku (vrátane prekážok, vojenských cvičení, prehliadok, závodov a väčších akcií parašutizmu mimo vyhlásených priestorov),
- výskyt alebo opravy dôležitých nedostatkov alebo závad v priestoroch prevádzkovej plochy,
- postavenie odstránenie alebo zmeny v prekážkach ovplyvňujúcich letovú prevádzku v priestoroch vzletu/stúpania, nepodareného priblíženia, priestoru priblíženia a dráhového pásu,
- únik rádioaktívnych materiálov alebo toxických chemikálií do atmosféry následkom nehody nukleárnych alebo chemických zariadení,
- výskyt,odstránenie alebo značná zmena nebezpečných podmienok spôsobených snehom, topiacim sa snehom, ladom, rádioaktívnym materiálom, toxickými látkami, usadzovaním vulkanického popola alebo vodou na pohybovej ploche,
- vypuknutie epidémií vyžadujúcich zmeny vo zverejnených požiadavkách na očkovanie a karanténne opatrenia.[12]

1.1 Typy správ NOTAM

Technologický posun a vývoj v leteckej doprave napreduje rýchlym tempom, čo sa odrazilo aj na správach NOTAM. Od vzniku a zavedenia do letovej prevádzky v nich bolo učinených mnoho zmien, ktoré ich formovali až do dnešnej podoby. Boli zavedené nové typy správ či už po technologickej alebo obsahovej stránke. Ich zavedenie prispelo k zjednodušeniu a zefektívneniu používania správ NOTAM.

1.1.1 NOTAM vs D–NOTAM

Správy NOTAM a Digital NOTAM (D–NOTAM) sú obsahovo rovnaké správy, rozdiel je v technológií akou sú prezentované.

Neštrukturované správy NOTAM sú správy v textovej podobe obsahujúce text písaný voľnou rečou. V rámci predletovej prípravy v dokumentácií Pre-flight Information Bulletins (PIB) sú posádke správy poskytnuté v papierovej podobe v množstve 10–50 strán, z ktorých je potrebné vyčítať všetky dôležité údaje. Často sa stáva, že text písaný voľnou rečou obsahuje v zápise chyby, nepresnosti alebo neobsahuje všetky potrebné informácie. V priemere 40-90% informácií obsiahnutých v PIB je prehliadnutých a nemajú na let pre ktorý sú určené vplyv.[3]

Obr. 1.1: Grafická vizualizácia D-NOTAMu[5]

Účel vzniku D–NOTAM je eliminovať problémy týkajúce sa správ NOTAM v textovej podobe, a ich nahradenie v celosvetovom merítku. D–NOTAM predstavuje digitálny, plne štrukturovaný dátový súbor. Informácie poskytnuté správou D–NOTAM sú určené k spracovaniu pomocou automatizovaného systému, ktorý ich

následne prezentuje buď v podobe textu alebo grafického formátu. Grafické zobrazenie dodáva pilotovi ale aj dispečerom priamy pohľad na stav objektu ktorý popisuje, bez potreby textovej dokumentácie. Na zavedenie D–NOTAM správy bolo potrebné vyvinúť model, ktorý bude popisovať pravidlá, pokyny, a spôsoby kódovania informácií z pôvodného textového formátu do digitálnej podoby vhodnej pre výmenu. K tomuto účelu sa využíva Aeronautical Information Exchange Model (AIXM), aktuálne vo verzii 5.1. Správa D–NOTAM môže byť použitá napríklad k vizualizácií aktuálneho stavu letiska tak ako je uvedené na obrázku 1.1.[5][7]

1.1.2 Špeciálne typy

V leteckej doprave sa opakovane vyskytujú nebezpečenstvá, ktorých zdrojom je určitá prírodná podmienka. Z tohto dôvodu vznikli špeciálne typy správ NOTAM, ktoré sa používajú výhradne vtedy, ak sa týkajú podmienky im prináležiacej. Tieto typy sú svojou štruktúrou a formou zápisu formulované tak, aby dokázali danú podmienku popísať čo najpresnejšie.

ASHTAM

Typ správy NOTAM nazývaný ASHTAM podáva informácie súvisiace s prebiehajúcou alebo očakávanou sopečnou činnosťou. Ak oblaky tvorené vulkanickým popolom vzniknuté pri erupcií predstavujú hrozbu pre letovú prevádzku, obsahuje ASHTAM informácie popisujúce ich polohu, rozsah, pohyb a nimi zasiahnuté letové cesty. ASHTAM sa skladá z položiek označených A) až K) ktoré sú abecedne zoradené a každej položke prináleží údaj o inej informácií. Maximálna doba platnosti je 24 hodín. Nový ASHTAM musí byť vydaný pri každej zmene úrovne varovného signálu. Úroveň varovného signálu sa udáva vo farbách a zapisuje sa do položky E). Úrovne sú nasledovné:

RED ALERT – je predpovedané nebezpečenstvo erupcie s pravdepodobnými významnými emisiami vulkanického popola do atmosféry/prebieha erupcia sopky s významnými emisiami vulkanického popola do atmosféry,

ORANGE ALERT – sopka javí zvýšenú pravdepodobnosť erupcie/ práve prebieha erupcia sopky so zanedbateľnými alebo žiadnymi emisiami vulkanického popola,

YELLOW ALERT – sopka vykazuje známky zvýšenej aktivity ktorá je nad rámec jej obvyklých hodnôt/vulkanická aktivita výrazne poklesla ale je potrebné ju naďalej sledovať z dôvodu možného opätovného vzostupu(po zmene z vyššej úrovne varovného signálu),

GREEN ALERT – sopka je v normálnom neaktívnom stave/vulkanická aktivita ustála a vrátila sa na svoju obvyklú hodnotu(po zmene z vyššej úrovne varovného signálu).[2][12]

BIRDTAM

BIRDTAM je špeciálny typ správy NOTAM, týkajúci sa nebezpečenstva tvoreného pohybom vtáctva najmä v nižších hladinách letového priestoru. Na rozdiel od ASHTAMu a SNOWTAMu nieje BIRDTAM oficiálnym termínom ICAO. Napriek tomu sú správy typu BIRDTAM rozpoznávané v European AIS Database (EAD) a majú svoju vlastnú AFTN adresu. Správa označená ako BIRDTAM, je formátovaná do štandardu North Atlantic Treaty Organization (NATO), vydávaná vojenskými službami. Správa BIRDTAM obsahuje informácie o čase platnosti, oblasti ktorej sa BIRDTAM týka, spodnej a hornej hranici priestoru v ktorom výstraha platí a úrovni intenzity pohybu vtáctva. Úroveň intenzity sa označuje číslom 0-8 pričom 8 je najvyššia možná. BIRDTAM sa vydáva len v prípade ak je hodnota rovná alebo vyššia ako 5. Významy možných označení intenzity sú uvedené v tabulke 1.1.[15][13]

Tab. 1.1: Označenia intenzity pohybu vtáctva v správe BIRDTAM

Označenie	**veľkosť intenzity**
5	pomerne veľká
6	veľká
7	veľmi veľká
8	extrémne veľká

SNOWTAM

Správa typu SNOWTAM sa používa na upozornenie o výskyte alebo ako informácia o odstránení nebezpečných podmienok na ploche letiska zapríčinených snehom, topiacim sa snehom, ľadom alebo stojacou vodou pôvodom zo snehu alebo ľadu. Maximálna doba platnosti správy SNOWTAM je 24 hodín. Nový SNOWTAM musí byť vydaný pri každej značnej zmene podmienok. Zoznam zmien, ktoré sú považované za dôvod k vydaniu novej správy SNOWTAM je uvedený v dokumente doc8126[10]. SNOWTAM sa skladá z abecedne zoradených položiek A) až T). Položka nemusí byť uvedená len v prípade, ak by bola prázdna a teda informácia ktorá by v nej mala byť uvedená nie je známa. Za vydanie správy SNOWTAM zodpovedá letisko.[10][8]

1.2 Skladba správy NOTAM

Správa NOTAM nesie údaje vždy len o jednej udalosti. Musí byť zostavená tak, aby bola jednoznačne dekódovateľná bez nutnosti využitia iného dokumentu. Z tohto dôvodu má každá z informácií obsiahnutých v správe NOTAM svoje presne dané umiestnenie v jednej z položiek, z ktorých sa správa skladá. Tieto položky majú presne stanovené poradie, v ktorom za sebou nasledujú. Položka nemusí byť v správe uvedená jedine v prípade, že v nej nie je obsiahnutá žiadna informácia.[2][12]

1.2.1 Séria a identifikátor

Každá správa NOTAM má pridelenú unikátnu sériu, ktorá sa skladá z veľkého písmena, štvormiestneho čísla, lomítka a dvojmiestneho čísla, ktoré označuje rok. Každá séria začína 1. januára číslom v tvare „0001“. Ak je vydaných viac sérií ako jedna, každá z nich musí byť identifikovaná veľkým písmenom tak, aby sa nezhodovali. K tomuto účelu sa využívajú písmena od A po Z s výnimkou písmen S a T. Za označením série nasleduje identifikátor, ktorý môže mať jeden z nasledujúcich tvarov: NOTAMN, NOTAMR, NOTAMC. Spôsob ich zápisu je uvedený na obrázku 1.2.Ich význam je nasledovný:

NOTAMN – písmeno N na konci značí anglické slovo „new“ ktoré sa prekladá ako nový. NOTAM správa označená týmto identifikátorom nesie nové informácie o predmete ktorého sa týka.

NOTAMR – písmeno R na konci značí anglické slovo „replacement“, ktoré sa prekladá ako náhrada. NOTAM správa označená týmto identifikátorom nahradzuje inú. Séria platnej správy ktorá je nahradená sa uvádza za identifikátor.

NOTAMC – písmeno C na konci značí anglické slovo „cancellation“, ktoré sa prekladá ako zrušenie. NOTAM správa označená týmto identifikátorom stornuje inú. Séria platnej správy ktorá je zrušená sa uvádza za identifikátor.[2]

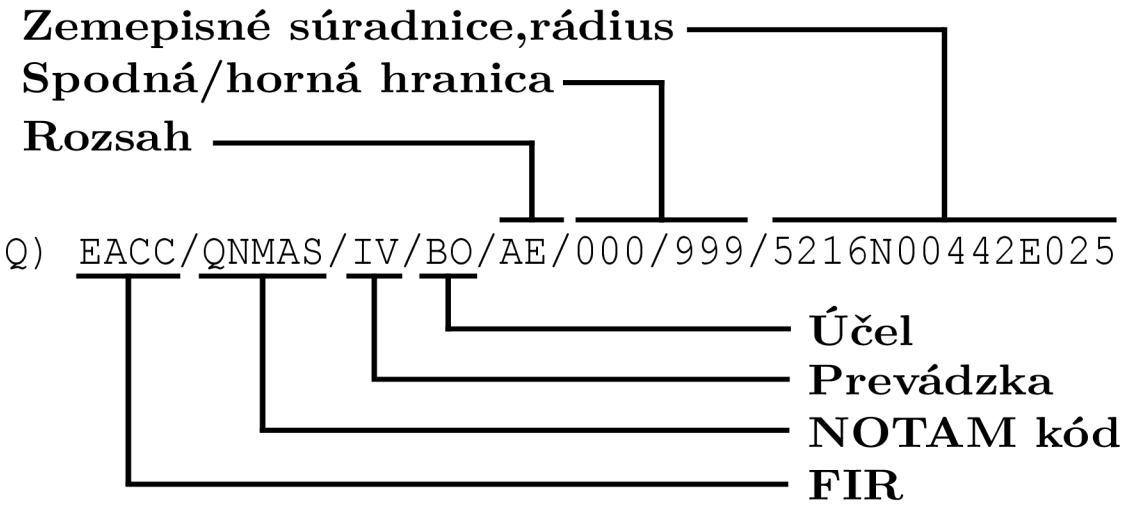
```
A0123/11 NOTAMN
A0124/11 NOTAMR A0123/11
A0125/11 NOTMAC A0124/11
```

Obr. 1.2: Spôsob zápisu série a identifikátoru

1.2.2 Položka Q)

Položka Q) je rozdelená do ôsmich polí, ktoré sú od seba oddelené lomítkom. Každé jedno z polí musí byť vyplnené. Definície jednotlivých polí a spôsob ich vyplnenia

je rozobratý v nasledujúcich podkapitolách. Ukážkový príklad položky s popisom jednotlivých polí je na obrázku 1.3[2][10]



Obr. 1.3: Skladba položky Q)

FIR

Názov pola Flight Information Region (FIR) predstavuje región leteckého priestoru v ktorom je poskytnutý Flight information service (FIS). Rozdelenie strednej Európy do jednotlivých FIR je na obrázku 1.5.

Rozlišujeme dva spôsoby zápisu pola FIR:

1. Ak je predmet informácie geograficky umiestnený vo vnútri jednej FIR, musí ICAO smerovacia značka odpovedať danej FIR. Ak je letisko umiestnené vo vnútri prekrývajúcej FIR iného štátu, musí toto pole obsahovať jej kód.
2. Ak je predmet informácie geograficky umiestnený vo viac ako v jednej FIR, musí pole FIR obsahovať prvé dve písmená ICAO smerovacej značky štátu vydávajúceho správu NOTAM, po ktorých nasleduje „XX“. V položke A) musia byť následne vymenované ICAO značky príslušných FIR alebo označenie štátu/nadvládnej organizácie zodpovednej za poskytovanie navigačnej služby vo viac než jednom štáte. Príklad oboch spôsobov zápisu je uvedený na obrázku 1.4.[2][12]

Q) EACC/	Q) EAXX/
A) EACC	A) EACC YUCC YUDD

Obr. 1.4: Ukážka zápisu FIR v položke Q)

Obr. 1.5: Rozdelenie strednej Európy podľa FIR[6]

NOTAM kód

Druhé pole položky Q) je tvorené piatimi veľkými písmenami. Prvé písmeno je vždy Q. Druhé a tretie písmeno udávajú informácie o predmete, štvrté a piate písmeno udávajú prevádzkový status predmetu alebo podmienky, ktorých sa správa NOTAM týka. Dvojpísmenné kódy pre predmety a podmienky sú uvedené v leteckom predpise L 8400 vydaným ministerstvom dopravy Českej republiky. Pravidlá pre správne používanie a kombinovanie týchto dvojpísmenných kódov sú obsiahnuté v Doc 8126, ktorý je vydaný pod ICAO. Ak sa kód pre predmet alebo prevádzkový status nenachádza ani v jednom z uvedených dokumentov, uvádza sa namiesto kódu „XX“, a to buď na druhé a tretie alebo na štvrté a piate miesto podľa toho, či sa jedná o predmet alebo prevádzkovú podmienku.[10][2]

Prevádzka

V tretom poli položky Q) sa uvádza typ prevádzky o ktorej nesie správa NOTAM informácie. V poli sa uvádza jedna z možností, ktoré sú uvedené v tabulke 1.2.[10][9]

Tab. 1.2: Možnosti a význam označenia v tretom poli položky Q)

Označenie	**Význam označenia**
I	Instrument Flight Rules (IFR) prevádzka
V	Visual Flight Rules (VFR) prevádzka
IV	kombinácia IFR a VFR
K	NOTAM je kontrolný zoznam platných NOTAM správ

Účel

Toto pole nesie informácie o účele, na ktorý je správa NOTAM určená, poprípade komu je adresovaná. Typ účelu je značený veľkými písmenami N, B, O, M alebo K a ich kombináciami. Povolené kombinácie sú: BO, NBO. Význam jednotlivých písmen je uvedený v tabulke 1.3.[10][2]

Tab. 1.3: Možnosti a význam označenia vo štvrtom poli položky Q)

Označenie	**Význam označenia**
N	určená k okamžitej pozornosti členov letovej posádky
B	prevádzkového významu určená pre PIB
O	týkajúca sa letovej prevádzky
M	nie je nevyhnutná pre predletovú prípravu, dodaná na vyžiadanie
K	NOTAM je kontrolný zoznam platných NOTAM správ

Rozsah

Štvrté pole udáva či sa predmet danej správy NOTAM spája s letiskom, traťou, navigačnou výstrahou alebo ide o správu ktorá je kontrolným zoznamom platných NOTAM správ.Taktiež udáva v ktorej kategórií bude správa prezentovaná v rámci PIB. Značenie je uvedené v tabulke 1.4.[2]

Spodná/horná hranica

Hodnoty týchto dvoch polí sa skladajú z trojmiestnych čísel, ktoré udávajú spodné a horné vertikálne obmedzenie daného priestoru. Hodnoty sú udávané v Flight Level (FL). Ak sa jedná o navigačné výstrahy uvedené hodnoty musia byť v súlade s hodnotami uvedenými v položkách F) a G). Hodnota spodnej hranice musí byť vždy menšia alebo rovná hranici hornej. Ak objektu správy NOTAM nenáleží informácia o spodnej alebo hornej hranici, uvádza sa „000“ ako spodná a „999“ ako horná hranica.[2][9]

Tab. 1.4: Možnosti a význam označenia piatého pola položky Q)

Označenie	**Význam označenia**
A	letisko
AE	letisko a trať
AW	letisko a výstraha
E	trať
W	navigačná výstraha
K	NOTAM je kontrolný zoznam platných NOTAM správ

Zemepisné súradnice/rádius

V poslednom poli položky Q) sa nachádzajú údaje o zemepisnej šírke a dĺžke s presnosťou na jednu minútu nasledované trojmiestnym číslom udávajúcim polomer v Nautical Mile (NM) tiež s presnosťou na jednu minútu. Zemepisné údaje udávajú približný stred kružnice, ktorej plocha zahrňuje celý dotknutý priestor. Ak sa správa NOTAM vzťahuje na celý UIR/FIR alebo k viac ako jednému UIR/FIR, uvádza sa hodnota rádiusu „999“.[10]

1.2.3 Položka A)

V položke A) sa podľa podľa ICAO Doc 7910 uvádza smerovacia značka letiska alebo FIR, v ktorom je umiestnené zariadenie, vzdušný priestor alebo podmienka o ktorej správa NOTAM nesie informácie. V jednej správe musí byť v položke A) uvedené len jedno letisko. Ak je potrebné, smerovacích značiek FIR/UIR môže byť uvedených viac. Ak ICAO smerovacia značka nie je známa, uvádza sa ICAO skratka štátu nasledovaná „XX“, a v položke E) sa následne uvedie meno v zrozumiteľnom tvare. Ak správa NOTAM nesie informácie týkajúce sa Global Navigation Satellite System (GNSS), uvádza sa smerovacia značka pridelená danej časti GNSS/všetkým častiam GNSS.[10][2][9]

1.2.4 Položka B)

Položka B) obsahuje desať miestne číslo, ktoré vyjadruje rok, mesiac, deň, hodinu a minútu v UTC. Správa NOTAM je platná od dátumu zverejnenia, t.j. dátum a čas vydania, avšak účinnosť v letovej prevádzke naberá až v čase uvedenom v položke B). Začiatok dňa sa uvádza hodnotou „0000“ v časti určenej pre hodiny a minúty. V prípade ak sa jedná o NOTAMR alebo NOTAMC je v tejto položke uvedený dátum a čas vydania správy NOTAM.[2][10][12]

1.2.5 Položka C)

Položka C) uvádza rok, mesiac, deň, hodinu a minútu v ktorej skončí platnosť správy NOTAM. Udáva sa v UTC, v tvare desať miestneho čísla. Ak sa jedná o správu neobmedzeného trvania, udáva sa namiesto čísel hodnota „PERM“. Koniec dňa sa označuje hodnotou „2359“ v časti určenej pre hodiny a minúty. Tvar „2400“ sa nepoužíva. Ak doba skončenia platnosti nie je dopredu presne určená, zadáva sa približná hodnota za ktorou nasleduje „EST“. Správa nesúca v položke C) označenie „EST“ musí byť pred uplynutím daného času zrušený alebo nahradený. V prípade ak sa tak nestane zostáva správa naďalej platná. Za vydanie podkladu o zrušení alebo nahradení je zodpovedný subjekt, ktorý vydal podklad na vydanie pôvodnej správy NOTAM. V prípade správy NOTAMC sa táto položka neudáva.[12][2]

1.2.6 Položka D)

V prípade že čas udalosti o ktorej správa nesie informácie nie je platný v celom intervale danom položkami B) a C), je v položke D) čas účinnosti správy NOTAM upresnený. V zápise sa vyskytujú čísla ale aj skratky vo forme veľkých písmen. Dni a Mesiace sa označujú skratkami v anglickom jazyku ktoré sa skladajú z troch písmen. V položke D) je možné uviesť presné rozmedzie dní a časov v ktorých bude správa NOTAM účinná v leteckej prevádzke. Napr. označenie „24H“ hovorí o platnosti celý deň, zápis „DAILY 0700-1000“ udáva opakovanie každý deň medzi 07:00–10:00 UTC. Platný zoznam všetkých skratiek a pravidiel zápisu informácie do položky D) je uvedený v Doc 8126 vydaným ICAO.[10][2]

1.2.7 Položka E)

Informácia v položke E) je v podobe textovej správy ktorá sa skladá z neštrukturovaných dát. Je v nej obsiahnutý dekódovaný NOTAM kód, doplnený podľa potreby ICAO skratkami, ukazovateľmi, identifikátormi, znakmi, názvami, frekvenciami, zemepisnými súradnicami, číselnými údajmi a textom písaným voľnou rečou. Ak je správa NOTAM určená k medzinárodnému rozosielaniu, text písaný voľnou rečou musí byť v angličtine. Text položky musí byť písaný stručne, zrozumiteľne, obsahovať všetky potrebné informácie pre bezpečné uskutočnenie letu, musí byť vhodný pre zaradenie do PIB a ideálne sa skladá z menej ako 300 znakov. Za vydanie a úpravu informácií poskytnutých príslušným zdrojom do konečnej podoby správy NOTAM zodpovedá Aeronautical Information Service (AIS).

Neštrukturovaný zápis je ťažko strojovo spracovateľný. Nieje presne určené poradie, ani syntax zápisu informácií. Tá istá informácia môže byť zapísaná viacerými spôsobmi, pričom každý z nich je správny. Záleží len na vydavateľovi správy

NOTAM, aký zápis zvolí. Touto problematikou sa zaoberá aj samostatná práca bližšie popísaná v kapitole 2. Jej cieľom je vytvorenie programu, ktorý spomedzi všetkých dát uvedených v položke E) vyberie práve tie, ktoré popisujú oblasť pre ktorú je správa platná a správne ich naformátuje do podoby vhodnej pre následné spracovanie.[10][2]

1.2.8 Položka F) a G)

Položky F) a G) označujú spodný a horný limit. Tieto položky sa zväčša používajú pre navigačné výstrahy alebo obmedzenia vzdušného priestoru a zvyčajne sú zahrnuté do PIB. Položky sa uvádzajú v zhodných jednotkách, a to buď v meter (m) alebo foot (ft). Pre označenie zeme alebo povrchu v položke F) sa používa značenie „GND“ alebo „SFC“. Pre označenie neobmedzeného horného limitu sa používa v položke G) označenie „UNL“. Zápis konkrétnej hodnoty výšky je v tvare čísla nasledovaného značkou použitej jednotky dĺžky za ktorou nasleduje jedna zo značiek „AMSL“/„AGL“, ktoré označujú výšku nad morom/pevninou. Ďalšia možnosť zápisu presnej hodnoty je formou zápisu v FL.[2]

1.2.9 Dekódovanie správy NOTAM

Tento odstavec je ukážkou príkladného dekódovania správy NOTAM uvedenej na obrázku 1.6.

```
A0623/11 NOTAMN
Q) EGXX/QRDCA/IV/NBO/W/000/400/5510N00520W050
A) EGTT/EGPX B)1104030730 C)1104281500
D) APR 03 07 12 21 24 AND 28 0730 to 1500
E) DANGER AREA DXX IS ACTIVE
F) GND G) 40000FT AMSL
```

Obr. 1.6: Ukážková správa NOTAM

Dekódovanie je nasledovné:

A0623/11 NOTAMN – A0623/11 značí sériu správy NOTAM. Jedná sa o správu vydanú v roku 2011. NOTAMN znamená že sa jedná o novú správu NOTAM.

Q) – EGXX - prvé dve písmená označujú ICAO smerovaciu značku Spojeného kráľovstva, posledné dve písmená značia že ICAO značky príslušných FIR, ktorých sa správa týka, budú jednotlivo vypísané v položke A),

- **QRDCA** – „RD“ značí že premetom správy je nebezpečná oblasť, „CA“ značí že oblasť je aktivovaná,
- **IV** – vzťahuje sa k obom prevádzkam IFR aj VFR,
- **NBO** – správa NOTAM je určený k okamžitej pozornosti členov letovej posádky a pre zahrnutie do PIB. Týka sa letovej prevádzky,
- **W** – jedná sa o navigačnú výstrahu[1],
- **000** – spodná hranica územia ktorého sa správa NOTAM týka je 0FL,
- **400** – horná hranica územia ktorého sa správa NOTAM týka je 400FL,
- **5510N00520W050** – oblasť ktorej sa správa NOTAM týka je kruh s polomerom 50 NM, ktorého stred sa nachádza na pozícií 55°10’ severnej zemepsinej šírky a 5°20’ západnej zemepisnej dĺžky,

A) – uvádza ICAO značky FIR ktorých sa správa týka, konkrétne sú to: **EGTT** - londýnsky FIR, **EGPX** - škótsky FIR,

B) – **1104030730** - správa NOTAM je patná od 3. 4. 2011 7:30 UTC,

C) – **1104281500** - správa NOTAM je platná do 28. 4. 2011 15:00 UTC,

D) – konkrétne dátumy dní a časové rozmedzie v ktorom je správa platná. Sú to dni 3., 7., 12., 21., 24. a 28. apríla v časovom rozmedzí od 7:30 do 15:00 UTC,

E) – nebezpečné územie je aktívne,

F) – **GND** - spodný limit je zem,

G) – **40000FT AMSL** - horný limit je 40000ft nad morom.[9][6]

[1]Pri navigačných výstrahách musia byť hodnoty spodnej/hornej hranice v položke Q) zhodné s hodnotami v položkách F) a G).

2 NÁVRH RIEŠENIA

Táto kapitola sa zaoberá implementáciou a stavbou algoritmov potrebných na spracovanie správ NOTAM do požadovanej štrukturovanej podoby, na vykreslovanie ich oblastí platnosti do mapy a na detekciu kolízie týchto oblastí s letovým plánom. Zameranie je najmä na správy popisujúce zloženú oblasť. Spracované sú len dáta popisujúce oblasť, ako napr. súradnice alebo hranice štátov. Ostatné informácie program nespracováva. Program bol vytvorený a testovaný na základe približne 100 správ NOTAM poskytnutých vedúcim práce, písaný je v programovacom jazyku Java a jeho funkcie sú nasledovné:

- správne otvorenie textového súboru obsahujúceho správy NOTAM,
- načítanie a overenie kompletnosti správ,
- správne parsovanie[1] každej jednej kompletnej správy NOTAM do štrukturovaného tvaru popisujúceho oblasť ktorej sa správa týka,
- vykreslenie oblastí správ NOTAM do mapy,
- detekcia kolízie oblastí správ s letovým plánom.

Daná funkcionalita programu je zabezpečená triedami, ktorých stavba a metódy sú bližšie popísané v nasledujúcich šiestich podkapitolách. Kompletný zdrojový kód tried, všetky využité prídavné súbory a spustiteľný Java Archive (JAR) súbor s programom je k dispozícií v prílohe A.

2.1 Trieda TextReader

Pomocou triedy `TextReader` je uskutočnené otvorenie textového súboru obsahujúceho správy NOTAM a ich načítanie. Syntax triedy bez definovaného obsahu tela metódy `loadNotams()` je na výpisku 2.1.

```
1  package Notam_decoder;
2
3  import java.io.BufferedReader;
4  import java.io.File;
5  import java.io.FileReader;
6  import java.io.IOException;
7
8  public class TextReader {
9    private File file;
10   private String key="";
11   private String value="";
12   private Parser parser=new Parser();
```

[1] V počítačovej vede označuje proces získania a spracovania informácií zo vstupu.

```
13
14   public TextReader(String cesta){
15     file=new File(cesta);
16     if(file.exists()==false){
17       System.out.println("Subor neexistuje");
18     }
19   }
20   public void loadNotams() throws IOException {...}
21 }
```

Výpis 2.1: Syntax triedy `TextReader`

Explicitný konštruktor triedy je definovaný s jedným parametrom dátového typu String, ktorý predstavuje cestu k súboru ktorý obsahuje správy NOTAM. Úlohou konštruktoru je overenie či súbor existuje a jeho otvorenie pomocou triedy `File`.

2.1.1 Metóda `loadNotams()`

Táto metóda s využitím tried `FileReader`, `BufferedReader` a cyklu while prechádza postupne po riadku celý textový súbor a do príslušných premenných ukladá hodnoty v správnom tvare. Identifikácia príslušných položiek je realizovaná pomocou podmienok if–else if. V prípade že riadok zodpovedá sérií a identifikátoru správy NOTAM uloží ho do premennej `key`. Ak program narazí na položku A) overí, či sa správa skladá z viacerých častí, ak áno do premennej `value` uloží informáciu o ktorú časť NOTAM správy sa jedná. Posledná podmienka overuje položku E)–ak na ňu program narazí, začne do premennej `value` ukladať každý riadok tejto položky. Následne sú premenné `key` a `value` predané parametrami funkcie `addNewNotam(String _key, String _value)` triedy `Parser` na ďalšie spracovanie. Hodnota premennej `value` sa nastaví na prázdny reťazec a znovu sa načítava nová správa NOTAM. Syntax bloku spracovania položky E) je na výpisku 2.2.

```
1  else if(line.startsWith("E)")==true){
2    while(line.startsWith("F)")==false && line.startsWith("END PART")
       ==false && line.startsWith("CREATED")==false){
3      value+=line;
4      value+=" ";
5      line=bufreader.readLine();
6    }
7
8    parser.addNewNotam(key, value);
9    value="";
10 }
```

Výpis 2.2: Spracovanie položky E)

2.2 Trieda Parser

Ako z názvu triedy vyplýva, úlohou triedy je naparsovanie obsahu položky E) popisujúcej oblasť platnosti NOTAM správy do požadovaného tvaru vhodného na ďalšie strojové spracovanie. Trieda obsahuje dve premenné `parsedNotams` a `unparsedNotams` typu `HashMap<String,String>` do ktorých sú ukladané NOTAM správy. Jednu premennú `mapOfCountries` typu `Map<String, String>` ktorej kľúče prezentujú rôzne tvary zápisu krajín v správe NOTAM a im náležiace hodnoty prezentujú zápis daných krajín v súbore použitom na vykreslovanie hraníc štátov. Ďalšie dve premenné `areaPrefixes` a `areaSuffixes` typu `String[]` sú využívané metódou `notamParser()` pri detekcií správ ktoré sa skladajú z viacerých oblastí. Požadovanú funkčnosť triedy zabezpečuje ďeväť metód. Metódy `printParsedNotams()` a `printNotams()` slúžia len na výpis naparsovaných/nenaparsovaných správ do konzole pomocou funkcie `System.out.println()` a na parsovanie nemajú vplyv. Metóda `createMapOfCountries()` slúži na inicializáciu premennej `mapOfCountries`, metódy `formatCoordinate(String original)` a `formatBorderInfo(String original)` a ich využitie je bližsie popísané v podkapitole 2.2.3 ktorá sa venuje syntaxu a sémantike funkcie `getAdaptedParsedNotams()`. Zvyšné dve metódy sú bližšie rozobrané v podkapitolách 2.2.1 a 2.2.2. Syntax triedy bez obsahu tiel metód je uvedený na výpisku 2.3.

```
1  package Notam_decoder;
2
3  import java.util.ArrayList;
4  import java.util.HashMap;
5  import java.util.List;
6  import java.util.Map
7
8  public class Parser {
9
10   private HashMap<String,String> unParsedNotams=new HashMap<String,
       String>();
11   private HashMap<String,String> parsedNotams=new HashMap<String,
       String>();
12   private static final Map<String, String> mapOfCountries =
       createMap();
13   private final static String[] areaPrefixes = {"AREA ","ZONE ","
       AREA-"} ;
14   private final static String[] areaSuffixes = {".  ",") AREA",".
       AREA"};
15
16   public void addNewNotam(String _key,String _value){...}
```

```
17
18    public void printNotams(){...}
19
20    public void printParsedNotams(){...}
21
22    public HashMap<String, String> getParsedNotams(){...}
23
24    private String formatCoordinate(String original){...}
25
26    private static Map<String, String> createMapOfCountries(){...}
27
28    private String formatBorderInfo(String original){...}
29
30    public HashMap<String,List<String>> getAdaptedParsedNotams(){...}
31
32    public void notamParser(){...}
33 }
```

Výpis 2.3: Syntax triedy `Parser` bez obsahu tiel metód

2.2.1 Metóda `addNewNotam(String _key, String _value)`

Táto metóda je volaná s dvomi parametrami z ktorých prvý označuje kľúč (séria a identifikátor NOTAM správy) a druhý hodnotu (obsah položky E)) ktorá má byť pridaná do premennej `unParsedNotams`. Metóda je tvorená podmienkami if–else if ktoré riadia pridávanie neparsovaných NOTAM správ do príslušnej premennej. V metóde je implemetovaná kontrola úplnosti správ NOTAM ktoré sa skladajú z viacerých častí tak, že parameter metódy `_key` sa porovná s množinou kľúčov premennej `unParsedNotams` a ak sa už medzi kľúčmi nachádza, program na základe označenia poradia časti spojí reťazec `_value` s reťazcom premennej `unParsedNotams` ktorý pripadá príslušnému kľúču `_key`. Týmto spôsobom program dokáže spojiť a následne spracovať NOTAM správy rozdelené na viaceré časti, avšak len v prípade že metóda je volaná s postupne po sebe vzostupne idúcimi časťami. Ak je správa uložená v premennej `unParsedNotams` neúplná, jej odpovedajúca hodnota reťazca obsahuje označenie „*****“.

2.2.2 Metóda `notamParser()`

Samotné konečné parsovanie správy NOTAM prebieha práve pomocou tejto metódy. Neštrukturované dáta položky E) obsiahnuté v premennej `unParsedNotams` sa pomocou metódy naparsujú a uložia do premennej `parsedNotams`. Parsovanie je realizované v niekoľkých cykloch úprav textu položky E).

V prvom for cykle je pomocou svojich kľúčov prechádzaná premenná `unParsedNotams`. Podmienkami je kontrolovaná úplnosť správy NOTAM a počet oblastí ktoré popisuje. Ak je správa úplná, uloží sa s príslušným kľúčom do lokálnej premennej `helpMap` typu HashMap¡String,String¿. Ak správa popisuje viac oblastí, v cykle je rozdelená podľa oblastí na prílušný počet častí a každá z nich je následne samostatne uložená do premennej `helpMap`. S premennou `unParsedNotams` sa už ďaľej nepracuje.

V druhom for cykle je pomocou svojich kľúčov postupne prechádzaná premenná `helpMap`. Na začiatku cyklu je reťazec pripadajúci danému kľúču rozdelený na základe množiny znakov {-, ,(,),?,;,’,:} a každá z častí je uložená do pola `helpArray` typu String. Následne je každá položka pola samostatne prechádzaná for cyklom, v ktorom je z položky končiacej bodkou alebo čiarkou odstránený posledný znak. Takto upravené položky sú následne uložené do lokálnej premennej `helpList` typu ArrayList¡String¿. Časť programu zodpovedná za tieto dve úpravy je zobrazená na výpisku 2.4.

```
1  helpArray=helpMap.get(key).split(" |\\-|\\(|\\)|\\?|\\;|\\’|\\:");
2
3  for(String x : helpArray){
4
5    if(x.length()>1&&x.charAt(x.length()-1)=='.'||x.length()>1&&x.
       charAt(x.length()-1)==','){
6      x=x.substring(0,x.length()-1);
7    }
8
9    if(x.isEmpty()!=true){
10     helpList.add(x);
11   }
12 }
```

Výpis 2.4: Rozdelenie a odstránenie prebytočných znakov položky E)

Hodnoty uložené v premennej `helpList` sú následne prechádzané for cyklom a podmienkami if–else if sú tieto hodnoty overované a ukladané do premennej `parsedNOTAM` typu String v požadovanom tvare. Do premennej `parsedNOTAM` sa dostanú len výsledné naparsované dáta ktoré popisujú oblasť platnosti správy NOTAM. Medzi tieto dáta patria súradnice, údaje popisujúce časti kružníc a údaje popisujúce hranice štátov. Nakoľko sú dáta obsiahnuté v položke E) neštrukturované a písané voľnou rečou, je k eliminácií spracovania nežiadúcich dát potrebných veľa podmienok. Parsovanie bere v úvahu aj správy NOTAM v ktorých sú vypísané dve oblasti pričom jedna nahradzuje druhú. Syntax spracovania slova OF je jeden z jednoduchších a jeho ukážka je uvedená na výpise 2.5.

Po dobehnutí cyklu je premenná `parsedNOTAM` uložená spolu s premennou `key` do premennej `parsedNotams` a premenné `helpList`, `parsedNOTAM` a `step` sa vynulujú pre využitie v ďalšom kroku nadradeného for cyklu prechádzajúceho premennú `helpMap`. Výsledný tvar naparsovaného NOTAMu je uvedený na obrázku 2.1.

```
1 else if(item.equals("OF")){
2
3   if(step==0){
4
5   }
6   else if(helpList.get(step-1).equals("ARC")||helpList.get(step-1).
     equals("CIRCLE")){
7     parsedNOTAM+=item+" ";
8   }
9 }
```

Výpis 2.5: Podmienky spracovania slova OF

```
W1255/15 NOTAMN AREA1
423033N0124959E
ARC OF CIRCLE CLOCKWISE RADIUS 7.5NM CENTRED ON 422534N0125101E
423125N0125724E
422929N0125516E
ARC OF CIRCLE COUNTER-CLOCKWISE RADIUS 5NM CENTRED ON 422534N0125101E
423033N0124959E

W1255/15 NOTAMN AREA2
423033N0124959E
ARC OF CIRCLE CLOCKWISE RADIUS 5NM CENTRED ON 422534N0125101E
422929N0125516E
422828N0125230E
423033N0124959E
```

Obr. 2.1: Ukážková naparsovaná správa NOTAM

2.2.3 Metóda `getAdaptedParsedNotams()`

Naparsované správy metódou `notamParser()` môžu obsahovať tú istú informáciu zapísanú v rôznom tvare. Zapríčinené je to tým, že metóda zo správy NOTAM iba vytiahne požadované informácie, ale neformátuje ich. Formátovanie naparsovaných

údajov zo správ do jednotných tvarov má na starosti práve funkcia `getAdaptedParsedNotams()`. Konkrétne sa jedná o formátovanie súradníc, častí kruhov a informácií popisujúcich hranice štátov. Funkcia pracuje ako for cyklus, ktorý prechádza pomocou kľúčov globálnej premennej `parsedNotams` jednotlivé naparsované správy NOTAM. Na začiatku cyklu sa každá správa rozdelí do poľa typu `String` na základe znaku nového riadku a toto pole je následne prechádzané ďalším vnoreným for cyklom. Na základe podmienok if–else if je v cyklu každý riadok naparsovanej správy rozpoznaný a podľa jeho obsahu naformátovaný. V každom kroku vnoreného for cyklu sa pridá naformátovaná informácia do premennej typu `ArrayList<String>` a po skončení tohoto cyklu sa do premennej typu `HashMap<String,List<String>>`, ktorú funkcia vracia, vloží hodnota, pričom kľúč je označenie série správy NOTAM a hodnota je premenná naplnená vnoreným for cyklom. Príklad ukážky naformátovanej správy z príkladu 2.1 je uvedený na obrázku 2.2 Spôsoby spracovania jednotlivých údajov sú nasledovné:

Ak sa jedná o súradnicu, podmienkou sa overuje či súradnica obsahuje desatinnú čiarku. Ak áno, súradnica sa predá ako argument funkcií `formatCoordinate(String original)` ktorá odstráni desatinné miesta a zabezpečí že všetky súradnice budú v rovnakom tvare. Ak súradnica desatinnú čiarku neobsahuje tak sa nemení.

Ak sa jedná o informáciu popisujúcu hranicu štátu, predá sa ako argument funkcií `formatBorderInfo(String original)` ktorá informáciu spracuje. Táto funkcia prechádza for cyklom kľúče globálnej premennej `mapOfCountries` a ak sa jedno zo slov popisujúcich informáciu o hranici zhoduje s nejakým z kľúčov, funkcia vráti hodnotu typu `String` v tvare „BORDER-hodnota“, kde hodnota predstavuje hodnotu z `mapOfCountries` pre daný kľúč. Ak sa v informácií o hranici nenachádza žiadny z kľúčov, funkcia vráti hodnotu v tvare „border info missing“. Správa NOTAM ktorej sa nepodarilo naformátovať informáciu o hranici nebude ďalej spracovaná.

Ak sa jedná o informáciu popisujúcu časť kruhu, najprv sa zistí či sa jedná o časť v smere alebo v protismere ručičkových hodiniek. Na základe toho sa do premennej `arcOfCircle` typu `String` uloží hodnota „INV“ alebo „NOT“. Následne je informácia rozdelená do pola typu `String` a hodnoty pola sú prechádzané for cyklom. Na základe podmienok sa v cykle vyhľadávajú informácie o polomere časti kruhu a o súradnici stredu kruhu. Ak sa nájde polomer, pridá sa na koniec premennej `arcOfCircle` v tvare „veľkosť v číslach nasledovaná jednotkou“. Ak sa nájde informácia o súradnici stredu kruhu, je v prípade potreby predaná funkcií `formatCoordinate(String original)` a následne pridaná na koniec premennej `arcOfCircle`, ktorá predstavuje výslednú naformátovanú informáciu o časti kruhu.

```
W1255/15 NOTAMN AREA1
423033N0124959E
NOT 7.5NM 422534N0125101E
423125N0125724E
422929N0125516E
INV 5NM 422534N0125101E
423033N0124959E

W1255/15 NOTAMN AREA2
423033N0124959E
NOT 5NM 422534N0125101E
422929N0125516E
422828N0125230E
423033N0124959E
```

Obr. 2.2: Ukážková naformátovaná správa NOTAM

2.3 Triedy reprezentujúce objekty

Triedy reprezentujúce objekty sú dve, konkrétne trieda `Notam` a trieda `FlightPlan`. Z názvov tried už vyplýva, že prvá trieda reprezentuje správu NOTAM, a druhá trieda reprezentuje letový plán.

2.3.1 Trieda Notam

Trieda `Notam` reprezentuje správu NOTAM pomocou troch premenných: `crossingFlightCorridor`, `coordinates` a `notamId`. Ich typy ako aj syntax triedy bez tela metód je uvedený na výpisku 2.6. Premenná `crossingFlightCorridor` nesie informáciu o kolízií oblasti správy NOTAM s letovým koridorom, `notamId` predstavuje označenie série a premenná `coordinates` predstavuje jednotlivé súradnice oblasti platnosti správy. Každá súradnica sa skladá zo zemepisnej šírky a zemepisnej dĺžky. Do premennej sú súradnice vkladané postupne, pričom sa vždy najprv vloží zemepisná šírka a po nej dĺžka. Trieda a jej metódy sú pomerne jednoduché. Obsahuje dva konštruktory, metódy get a set pre každú premennú a metódu `addCoordinate(double latCoordinate, double lonCoordinate)` ktorá slúži na pridanie súradnice do premennej `coordinates`.

```
1 package Notam_decoder;
2
3 import java.util.ArrayList;
4 import java.util.List;
```

```
5
6  public class Notam {
7
8    private boolean crossingFlightCorridor = false;
9    private List<Double> coordinates = new ArrayList<>();
10   private String notamId;
11
12   public Notam(){...}
13
14   public Notam(String notamId){...}
15
16   public boolean getCrossingFlightCorridor(){...}
17
18   public void setCrossingFlightCorridor(boolean
       crossingFlightPathCorridor){...}
19
20   public List<Double> getCoordinates(){...}
21
22   public void setCoordinates(List<Double> coordinates){...}
23
24   public String getNotamId(){...}
25
26   public void setNotamId(String notamId){...}
27
28   public void addCoordinate(double latCoordinate, double
       lonCoordinate){...}
29 }
```

Výpis 2.6: Syntax triedy Notam

2.3.2 Trieda FlightPlan

Trieda `FlightPlan` reprezentuje letový plán. Obsahuje dve premenné `pathCoordinates` a `corridorCoordinates`. Obe premenné sú typu `List<Double>` a ako z ich názvov vyplýva popisujú súradnice letovej cesty a letového koridoru. Podobne ako u premennej `coordinates` triedy `Notam` 2.3.1, aj v tomto prípade sa do premenných ukladajú súradnice rozdelené na zemepisnú šírku a zemepisnú dĺžku, pričom sa vždy ako prvá vkladá zemepisná šírka. Trieda taktiež obsahuje konštruktor, metódy get a set pre premenné, metódy na pridanie súradnice do premenných a tiež metódu na odstránenie súradnice z premennej `pathCoordinates`. Syntax triedy bez tiel metód je na výpisku 2.7.

```
1 package Notam_decoder;
2
3 import java.util.ArrayList;
```

```
4  import java.util.List;
5
6  public class FlightPlan {
7    private List<Double> pathCoordinates;
8    private List<Double> corridorCoordinates;
9
10   public FlightPlan(){..}
11
12   public List<Double> getPathCoordinates(){...}
13
14   public void setPathCoordinates(List<Double> pathCoordinates){...}
15
16   public List<Double> getCorridorCoordinates(){...}
17
18   public void setCorridorCoordinates(List<Double>
       corridorCoordinates){...}
19
20   public void addPathCoordinate(double latCoordinate, double
       lonCoordinate){...}
21
22   public void addCorridorCoordinate(double latCoordinate, double
       lonCoordinate){...}
23
24   public void removePathCoordinate(int pointNumber){...}
25 }
```

Výpis 2.7: Syntax triedy `FlightPlan`

2.4 Trieda BorderComputer

`BorderComputer` je jedna z dvoch tried ktoré obsahujú funkcie s logikou výpočtu oblasti správy NOTAM. Trieda zabezpečuje logiku výpočtu časti oblasti, ktorá sa viaže k hranici určitého štátu. V triede sa využívajú triedy a funkcie definované v knižnici GeoTools, ktorá slúži aj na prácu so súborom formátu shapefile[2]. Tento formát je určený najmä na priestorovú reprezentáciu rôznych geografických prvkov, akými sú napríklad jazerá, národné parky, moria alebo štáty. Prvky sú väčšinou reprezentované množinou súradníc, ktoré opisujú ich geografickú polohu a doplňujúcimi informáciami podla objektu ktorý reprezentujú. V tejto práci a konkrétne v tejto triede je použitý súbor ktorý obsahuje prvky reprezentujúce štáty a množinou súradníc popisuje ich hranice. Nakoľko je jedna z vlastností programu jeho spustiteľnosť

[2]Konkrétny súbor formátu shapefile využitý v tejto práci je získaný z <`http://www.naturalearthdata.com/http//www.naturalearthdata.com/download/10m/cultural/ne_10m_admin_0_countries.zip`>

bez potreby prídavných súborov, sú tieto súbory zabalené priamo v spustiteľnom JAR súbore. V konštruktore tejto triedy je prevedené načítanie potrebných súborov k práci so súborom typu shapefile, a následne sú na disku vytvorené ich dočasné kópie tak, aby bol program schopný so súbormi pracovať. Súbor typu shapefile je v konštruktore otvorený a načítaný do objektu typu `ShapefileDataStore`. Okrem konštruktoru, ktorého implementácia je zobrazená na výpisku 2.8, obsahuje trieda dve metódy ktoré implementujú výpočet. Tieto metódy sú bližšie popísane v odstavcoch 2.4.1 a 2.4.2.

```
1  public BorderComputer(){
2      try{
3      tempDir = Files.createTempDirectory("").toFile();
4      tempDir.deleteOnExit();
5      InputStream streamShp=getClass().getClassLoader().
       getResourceAsStream("shapefile.shp");
6      InputStream streamShx=getClass().getClassLoader().
       getResourceAsStream("shapefile.shx");
7      InputStream streamDbf=getClass().getClassLoader().
       getResourceAsStream("shapefile.dbf");
8        FileShp = new File(tempDir+"/shapefile.shp");
9        FileShp.deleteOnExit();
10       FileShx = new File(tempDir+"/shapefile.shx");
11       FileShx.deleteOnExit();
12       FileDbf = new File(tempDir+"/shapefile.dbf");
13       FileDbf.deleteOnExit();
14       FileUtils.copyInputStreamToFile(streamShp,FileShp);
15       FileUtils.copyInputStreamToFile(streamShx,FileShx);
16       FileUtils.copyInputStreamToFile(streamDbf,FileDbf);
17
18       dataStore = new ShapefileDataStore(FileShp.toURI().toURL());
19     }catch(Exception e){
20         Gui.log(e.getMessage());
21         e.printStackTrace();
22     }
23   }
```

Výpis 2.8: Syntax konštruktoru triedy `BorderComputer`

2.4.1 Metóda `computeBorderCoordinates( ... )`

Celý tvar hlavičky tejto metódy: `public List<Double> computeBorderCoordinates(double lat1, double lon1, double lat2, double lon2,String country, List<Double> coordinates)`. Prvé štyri argumenty funkcie predstavujú počiatočnú

a konečnú súradnicu časti oblasti idúcej pozdĺž hranice. Piaty argument predstavuje názov krajiny a `coordinates` je množina súradníc ku ktorej sa majú pridať nové vyrátané súradnice. V tejto triede sa najprv pomocou objektov typov `FeatureSource`, `FeatureCollection`, `DataSource` a ich funkcií získa kolekcia prvkov reprezentujúcich jednotlivé krajiny zo súboru typu shapefile. Táto kolekcia sa uloží do objektu `iterator` typu `FeatureIterator` ktorého metóda `hasNext()` nám umožňuje while cyklom prechádzať jednotlivé prvky. V tomto cykle sa vždy na začiatku načíta prechádzaný prvok do objektu typu `Feature`. Pomocou objektu tohoto typu sme schopný načítať z prvku jeho informácie. V if podmienke je porovnávaný názov krajiny zadaný ako argument funkcie s aktuálnym názvom krajiny uvedeným v prvku ktorý je prechádzaný cyklom while. V prípade zhody sa do objektu typu `GeometryAttribute` načíta geometrický popis daného štátu. Z tohto popisu, ktorý predstavuje množinu bodov opisujúcu hranicu štátu, sa odstránia nadbytočné znaky a uloží sa do premennej `points` typu `String`. Metóda následne príkazom return vracia volanie funkcie `computePoints( ... )` ktorej predáva ako argumenty získanú množinu bodov `points`, počiatočnú a konečnú súradnicu a množinu súradníc ktoré boli funkcií dané v podobe argumentov. V prípade že v podmienke if nenájde funkcia zhodu, vracia hodnotu null ako chybu.

2.4.2 Metóda `computePoints( ... )`

Celý tvar hlavičky tejto metódy: `private List<Dou ble> computePoints(String points, double lat1, double lon1, double lat2, double lon2, List<Double> coordinates)`. Prvý z argumentov predstavuje množinu bodov opisujúcich hranicu štátu. Ďalšie argumenty predstavujú počiatočnú a konečnú súradnicu časti hranice štátu ktorá má byť pridaná k súradniciam predaným v poslednom argumente. Táto funkcia vracia novú množinu súradníc zloženú z pôvodných a nových – popisujúcich časť oblasti idúcej pozdĺž hranice. V prvom kroku funkcia rozdelí množinu bodov do pola `helpArray`. Toto pole je následne prechádzané for cyklom, ktorého úlohou je nájsť index najbližšej súradnice k počiatočnej a ku konečnej súradnici v tomto poli. Tieto indexy sú uložené v premennej `closestPoints` typu `HashMap<String,Integer>()`. Kľúč pre index súradnice najbližšej k počiatočnej je označený ako „point1“, kľúč pre index súradnice najbližšej ku konečnej je označený ako „point2“. Po tomto kroku máme nájdené dva indexy v poli súradníc, pričom tieto dva indexy ohraničujú množinu indexov na pozícií ktorých sú súradnice, ktoré majú byť pridané do zoznamu súradníc vrátane dvoch krajných. Taktiež vieme ktorý index označuje súradnicu najbližšiu k počiatočnej a ktorý ku konečnej. To, že súradnice budú pridané do zoznamu súradníc v správnom poradí je zabezpečené následnými if–else podmienkami, ktoré kontrolujú hodnotu indexov uložených v pre-

mennej `closestPoints`, a na základe toho je vykonaný jeden z dvoch for cyklov ktorý pridá súradnice do premennej `coordinates` v správnom poradí. Ukážka implementácie týchto podmienok a následne zvolených for cyklov je na výpisku 2.9. Posledný krok funkcie je príkaz return ktorým funkcia vráti množinu súradníc s novými popisujúcimi oblasť popri hranici štátu.

```
if(closestPoints.get("point1")  <= closestPoints.get("point2")){
  for(int i=0;i<helpArray.length;i++){
    if(i >= closestPoints.get("point1") && i<=closestPoints.get("
   point2")){
      helpArray[i] = helpArray[i].trim();
      lonLat =helpArray[i].split(" ");
      lat = Double.parseDouble(lonLat[1]);
      lon = Double.parseDouble(lonLat[0]);
      coordinates.add(lat);
      coordinates.add(lon);
    }
  }
}
else if(closestPoints.get("point1")  > closestPoints.get("point2"))
   {
  for(int i=helpArray.length - 1;i>=0;i--){
    if(i >= closestPoints.get("point2") && i<=closestPoints.get("
   point1")){
      helpArray[i] = helpArray[i].trim();
      lonLat =helpArray[i].split(" ");
      lat = Double.parseDouble(lonLat[1]);
      lon = Double.parseDouble(lonLat[0]);
      coordinates.add(lat);
      coordinates.add(lon);
    }
  }
}
```

Výpis 2.9: Syntax zvolenia správneho for cyklu pre pridanie súradníc popisujúcich hranicu

2.5 Trieda AreaComputer

Trieda `AreaComputer` obsahuje metódy, ktoré sú zodpovedné za celkový výpočet celej oblasti správy NOTAM ale aj oblasti letového koridoru a výpočet kolízie letového plánu s oblasťou správy NOTAM. Trieda obsahuje len jednu statickú premennú–objekt typu `BorderComputer`. Tento objekt je využitý pri výpočte oblasti ktorej časť je súbežná s hranicou určitého štátu. Trieda obsahuje desať metód, z čoho

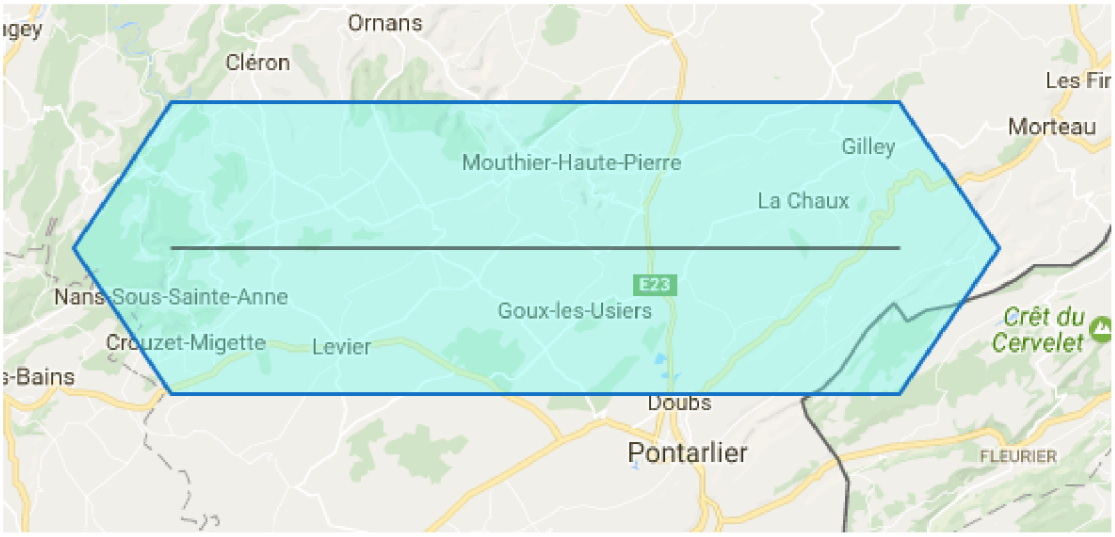
tri sú hlavné výpočetné a ostatných sedem slúži ako pomocné metódy pri výpočte. Všetky metódy sú verejné a statické, nakoľko môžu byť nezávisle od objektu použité v inej triede na určitý výpočet. Sedem pomocných metód slúži napríklad na výpočet súradníc časti kruhu, výpočet vzdialenosti dvoch súradníc alebo formátovanie súradnice z textového formátu do typu `Double`. Ostatné tri hlavné metódy sú popísane ďalej v práci v odstavcoch 2.5.1, 2.5.2 a 2.5.3.

2.5.1 Metóda `computeAreaIntersection( ... )`

Celý tvar Hlavičky metódy je nasledovný: `public static boolean computeAreaIntersection(List<Double> area1Coordinates, List<Double> area2Coordinates)`. Z názvu metódy vyplýva, že sa jedná o metódu určenú na výpočet kolízie letového koridoru s oblasťou správy NOTAM. Funkcia dostane v podobe argumentov dve množiny súradníc, pričom každá množina predstavuje jednu oblasť. V prvej časti funkcia kontroluje, či nejaký z bodov ktorými je definovaná druhá oblasť leží vo vnútri oblasti definovanej prvým argumentom funkcie. Pomocou for cyklu zo súradníc argumentu `area1Coordinates` vytvorí objekt `area1` typu `Area`. Následne sú súradnice argumentu `area2Coordinates` prechádzané for cyklom, pričom v každom kroku sa zo súradnice vytvorí objekt typu `Point2D`. Zároveň sa v každom kroku skontroluje, či oblasť premennej `area1` neobsahuje aktuálnu súradnicu. Ak oblasť súradnicu obsahuje, funkcia vráti hodnotu „true“. Ak nie výpočet funkcie pokračuje. V druhej časti funkcia kontroluje, či sa dané oblasti neprekrývajú bez toho, aby body ktorými sú definované ležali vo vnútri druhej oblasti. Táto kontrola je naimplementovaná pomocou dvoch for cyklov, pričom jeden je vnorený. Prvý for cyklus prechádza súradnice prvého argumentu funkcie a vytvára z nich objekty typu `Point2D` z ktorých následne vytvorí objekt typu `Line2D`. Vnorený for cyklus funguje tak isto s tým rozdielom, že prechádza súradnice druhého argumentu funkcie. Vo vnorenom for cykle je podmienka, ktorá kontroluje či sa aktuálne objekty typu `Line2D`, predstavujúce úsečky, krížia. Táto kontrola je prevedená pomocou funkcie `intersectsLine(Line2D l)` triedy `Line2D`. V prípade že existuje priesečník týchto úsečiek, funkcia vráti hodnotu „true“. Ak sa v daných for cykloch skontrolujú všetky možné kombinácie úsečiek tvoriacich jednotlivé oblasti bez detekcie ich prieseku, tieto oblasti sa považujú za oblasti bez vzájomnej kolízie a funkcia vracia hodnotu „false“.

2.5.2 Metóda `computeFlightCorridor(List<Double> flightPath)`

Pomocou tejto metódy dokáže program vyrátať zo zadaných súradníc letovej cesty letový koridor. Funkcia sa dá rozdeliť na dva celky. V prvom celku sa pomocou for cyklu prechádza zoznam súradníc letovej cesty ktorý je argumentom funkcie. Pre každý dvojicu za sebou idúcich súradníc sa vyráta oblasť v tvare šesťuholníka, pričom vzdialenosť letovej cesty a tohoto šesťuholníka je nastavená tak, aby predstavovala v mape štyri NM do každej strany. Tento šesťuholník je reprezentovaný objektom `Path2D`, z ktorého sa následne vytvorí objekt typu `Area` a ten sa pomocou funkcie `add(Area rhs)` triedy `Area` pridá ku globálnej premennej `area` tejto funkcie, ktorá predstavuje celý letový koridor zložený z jednotlivých šesťuholníkov. Ukážka vykreslenia letového plánu definovaného len dvomi súradnicami, čo predstavuje jeden šesťuholník, je na obrázku 2.3.



Obr. 2.3: Vykreslenie letového plánu definovaného dvoma súradnicami

Nakoľko má táto funkcia návratovú hodnotu typu `List<Double>`, je potrebné výsledný objekt typu `Area` zložený zo všetkých šesťuholníkov previesť na jednotlivé súradnice ktoré budú opisovať jeho tvar. Tento prevod je zabezpečený pomocou objektu `pathIterator` typu `PathIterator`, ktorý je nainicializovaný pomocou funkcie `getPathIterator(AffineTransform at)` objektu `area`. Pomocou cyklu while sú prechádzané a ukladané jednotlivé súradnice objektu `pathIterator` do premennej typu `List<Double>` ktorá predstavuje návratovú hodnotu funkcie. Po prejdení a uložení všetkých súradníc opisujúcich vypočítaný letový koridor funkcia príkazom return vráti získané súradnice.

2.5.3 Metóda computeNotams(HashMap <String, List <String>> parsedNotams)

Táto metóda kombináciou využitia všetkých pomocných metód tejto triedy a pomocou metód triedy `BorderComputer` vracia zoznam objektov triedy `Notam`, ktoré majú v premennej `coordinates` definované súradnice popisujúce oblasť ktorej sa daná správa týka. Argumentom funkcie je objekt typu `HashMap <String,List<String>>` ktorého kľúče sú jednotlivé série správ NOTAM a im náležiace hodnoty sú naparsované a naformátované správy NOTAM. Funkcia na začiatku overí či argument nie je prázdny, ak nie pokračuje. For cyklom sa prechádza zoznam kľúčov argumentu funkcie. Každý kľúč prináleží jednej správe NOTAM, a preto sa pri každej jednej iterácií vytvorí danému kľúču náležiaci objekt typu `Notam` označený ako `newNotam`. Na konci každého cyklu sa objekt `newNotam` pridá do premennej `returnValue` typu `List<Notam>` ktorá predstavuje návratovú hodnotu funkcie. V tele for cyklu prechádzajúceho kľúče argumentu funkcie sa ako prvá nachádza podmienka overujúca, či daná oblasť správy NOTAM nie je iba jednoduchý kruh. V prípade že sa jedná len o oblasť popísanú ako jednoduchý kruh, objektu `newNotam` sa nastaví premenná `coordinates` pomocou funkcie `setCoordinates(List<Double> coordinates)` na požadované súradnice. Tieto súradnice sú vyrátané pomocnou funkciou `computeCirclePoints(double lat, double lon, String distance)` ktorej argumentmi sú súradnica a polomer kruhu. premenná `newNotam` sa následne uloží do premennej `returnValue` a príkazom continue sa preskočí na iteráciu pre ďalší z kľúčov argumentu funkcie `parsedNotams`. V prípade že sa nejedná o správu NOTAM s oblasťou v podobe jednoduchého kruhu, funkcia pokračuje vnoreným for cyklom. V tomto cykle sú na základe kľúču ktorý je aktuálne v nadradenom for cykle prechádzané jednotlivé údaje správy NOTAM, ktoré sú v podobe objektu `List<String>`. Údajmi môže byť buď súradnica, časť kruhu alebo informácia o časti oblasti idúcej pozdĺž štátnej hranice.

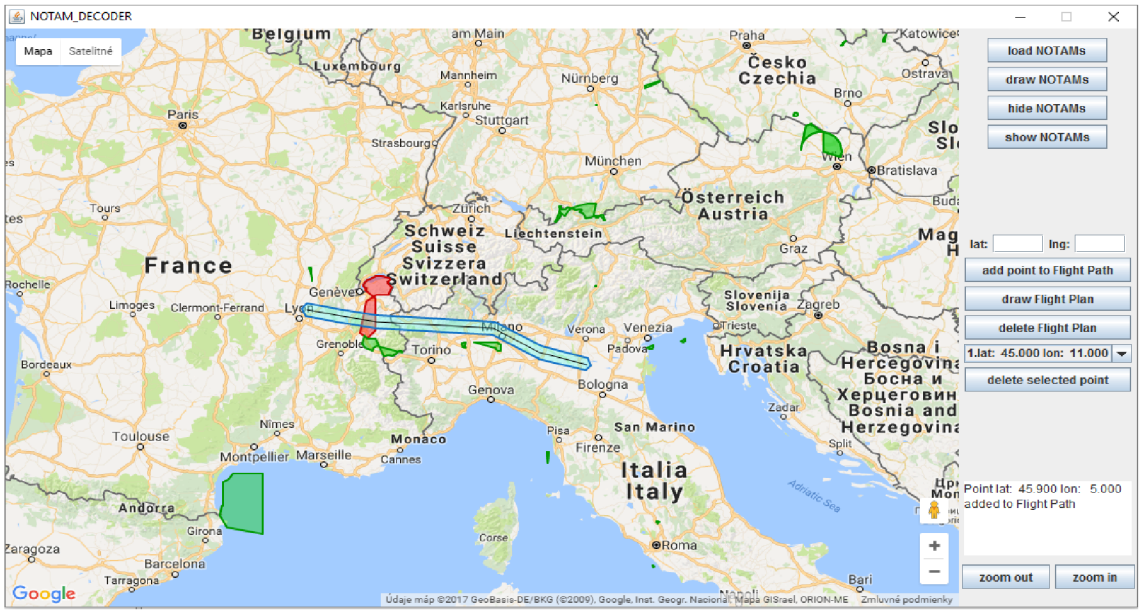
V prípade že sa jedná o údaj popisujúci časť oblasti idúcej pozdĺž štátnej hranice, funkcia zavolá funkciu `computeBorderCoordinates( ... )` ktorá vráti nové súradnice popisujúce oblasť správy aj s pridanými súradnicami definujúcimi časť oblasti idúcej pozdĺž štátnej hranice. Následne sa objektu `newNotam` nastaví premenná `coordinates` na novú aktuálnu hodnotu.

V prípade že sa jedná o údaj popisujúci časť kruhu, časti popisujúce túto informáciu sa uložia do pola na základe znaku medzery. Tieto informácie sú stred kružnice, jej polomer a informácia o tom či je časť kruhu kreslená v smere alebo v protismere hodinových ručičiek. K získaniu súradníc popisujúcich časť kruhu je potrebné získať z predchádzajúceho spracovaného údaju súradnicu označujúcu začiatok časti kruhu a z nasledujúceho údaju súradnicu popisujúcu koniec časti

kruhu. Ak je údaj popisujúci časť kruhu posledný, tak sa ako súradnica označujúca koniec časti kruhu vezme súradnica z prvého údaju. Podľa toho, či je kružnica v smere alebo v proti smere hodinových ručičiek je volaná jedna z pomocných metód `computeArcPoints(double arcPoints[],List<Double> coordinates)`/`computeInvArcPoints(double arcPoints[],List<Double> coordinates)` ktorej sú predané argumenty popisujúce začiatok, koniec a stred časti kruhu ako aj aktuálny zoznam súradníc popisujúcich oblasť správy. Tieto funkcie vrátia množinu pôvodných súradníc ku ktorej sú pridané súradnice popisujúce časť kruhu. Následne sa do premennej `coordinates` objektu `newNotam` uložia nové hodnoty súradníc a pokračuje sa v cykle.

V prípade že sa jedná o údaj popisujúci súradnicu, súradnica sa naformátuje a podmienkami sa overí, že súradnica nie je posledná a zároveň zhodná s prvou/súradnica sa už medzi súradnicami nachádza lebo v spracovaní predchádzajúceho údaju bola po ňu vykreslená časť kruhu. Ak ani jedna z týchto podmienok nie je pravdivá, súradnica sa pomocou funkcie `addCoordinate(double latCoordinate, double lonCoordinate)` objektu `newNotam` pridá na koniec zoznamu súradníc ktoré popisujú jeho oblasť platnosti.

2.6 Trieda Gui

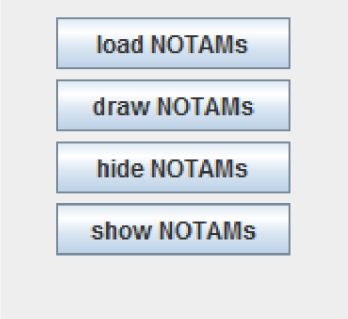


Obr. 2.4: Grafické rozhranie programu

Ako už z názvu triedy vyplýva, jedná sa o triedu reprezentujúcu grafické užívateľské

rozhranie. Trieda obsahuje najmä objekty ktoré prezentujú jednotlivé komponenty užívateľského rozhrania ale taktiež aj objekty reprezentujúce zoznam správ NOTAM alebo aj letový plán. Táto trieda spája všetky ostatné triedy do jedného funkčného celku ovládateľného užívateľom. Nastavenie vzhľadu a umiestnenia jednotlivých komponentov má na starosti metóda `Initialize()` ktorá je volaná v konštruktore triedy. Na obrázku 2.4 je vidieť rozloženie komponentov a celkový vzhľad užívateľského rozhrania. Zobrazenie a práca s mapou je zabezpečená pomocou komponentov typu `Browser` a `BrowserView`[3] ktoré zobrazujú súbor typu HTML, ktorý je priložený v prílohe A. V pravej časti rozhrania je umiestnená komponenta typu `JPanel` v ktorej sú umiestnené tri ďalšie komponenty typu `JPanel` pomenované ako `rightTopPanel`, `rightMidPanel` a `rightbottomPanel`. Z ich pomenovaní je zrejmé že sa jedná o spodný, stredný a vrchný panel. Každý z panelov, vrátane funkcionality komponentov ktoré obsahuje, je bližšie popísaný v odstavcoch 2.6.1, 2.6.2 a 2.6.3.

2.6.1 Panel rightTopPanel



Obr. 2.5: Rozloženie komponentov v panely `rightTopPanel` uživateľského rozhrania

Panel `rightTopPanel` obsahuje tlačidlá na prácu so správami NOTAM, pričom tento panel má usporiadanie komponentov nastavené ako `BoxLayout`. Konkrétne sa jedná o tlačidlá označené ako load NOTAMs, draw NOTAMs, hide NOTAMs a show NOTAMs. Funkcionalita tlačidiel odpovedá ich názvom. Posledné dve z uvedených tlačidiel pomocou funkcie `executeJavaScript(String JavaScriptCommand)` objektu typu `Browser` zavolajú im náležiacu funkciu definovanú v HTML súbore (príloha A) a pomocou nej buď skryjú alebo zobrazia vykreslené správy NOTAM.

Tlačidlo load NOTAMs zabezpečuje načítanie súboru so správami NOTAM,ich následné naparsovanie a uloženie do globálnej premennej `actualNotams`. Výber súboru je implementovaný využitím objektu typu `JFileChooser` pomocou ktorého je

[3]Tieto dve triedy sú súčasťou knižnice JxBrowser vyvýjanej spoločnosťou TeamDev, ktorá mi po vzájomnej dohode poskytla akademickú licenciu s platnosťou do 28. 6. 2017.

užívateľ schopný vybrať súbor so správami NOTAM. Tento súbor je následne spracovaný funkciami `loadNotams()`, `notamParser()` a `getAdaptedParsedNotams()` objektov tried `TextReader` a `Parser`. Návratová hodnota poslednej z uvedených funkcií sa ukladá do premennej `actualNotams`.

Tlačidlom draw NOTAMs je užívateľ schopný vykresliť naparsované správy NOTAM do mapy. V prvom kroku funkcia zmaže aktuálne vykreslené oblasti a následne naplní globálnu premennú `notams` typu `List<Notam>` hodnotami pomocou statickej funkcie `computeNotams(HashMap <String,List<String>> parsedNotams)` triedy `AreaComputer`. Premenná `notams` je následne prechádzaná for cyklom a pre každý objekt triedy `Notam` je vyrátaná jeho premenná `crossingFlightCorridor`. Následne je realizované vykreslenie správ NOTAM do mapy pomocou funkcie `executeJavaScript(String JavaScriptCommand)` objektu typu `Browser`, ktorej argumentmi sú JavaScript výrazy využívajúce funkcie a premenné definované v zobrazovanom HTML súbore a v Google Maps API[4]. Oblasti správ NOTAM ktoré sa krížia s letovým plánom sa vykreslia červenou farbou, ostatné zelenou viď. obrázok 2.4. Syntax funkcie ktorá je zavolaná po kliknutí na tlačidlo draw NOTAMs je na výpisku 2.10.

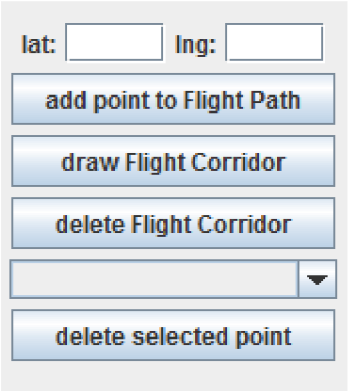
```
1  private void drawNotams(ActionEvent e){
2      if(actualNotams.isEmpty()){
3        log("No NOTAMS loaded.");
4        return;
5      }
6      if(!notamsPrinted){
7        hideNotams(e);
8        browser.executeJavaScript("notams.length=0");
9
10       notams=AreaComputer.computeNotams(actualNotams);
11       for(Notam notam : notams){
12         if(flightPlan.getCorridorCoordinates().size()>0){
13           notam.setCrossingFlightCorridor(AreaComputer.
     computeAreaIntersection(flightPlan.getCorridorCoordinates(),
     notam.getCoordinates()));
14         }
15         for(int i = 0 ;i<notam.getCoordinates().size();i+=2){
16           browser.executeJavaScript("Point={lat: "+notam.
     getCoordinates().get(i).toString()+", lng: "+notam.
     getCoordinates().get(i+1).toString()+"}");
17           browser.executeJavaScript("arcPts.push(Point)");
18         }
19         browser.executeJavaScript("piePoly=new google.maps.Polygon
     ({"
20             +"paths: [arcPts],"
```

[4]Application programming interface (API) označuje zbierku funkcií a tried (ale aj iných programov), ktoré určujú akým spôsobom sa majú funkcie knižníc volať zo zdrojového kódu programu.[1]

```
21              +"strokeColor: \"#009900\","
22              +"strokeOpacity: 1,"
23              +"strokeWeight: 2,"
24              +"fillColor: \"#00CC00\","
25              +"fillOpacity: 0.4,"
26              +"map: map});");
27          if(notam.getCrossingFlightCorridor()){
28            browser.executeJavaScript("piePoly.setOptions({
    strokeColor: \"#CC0000\",fillColor: \"#FF0000\"})");
29          }
30          browser.executeJavaScript("notam = {identifier:\""+notam.
    getNotamId()+"\",polygon:piePoly}");
31          browser.executeJavaScript("notams.push(notam)");
32          browser.executeJavaScript("arcPts.length=0");
33        }
34      }
35      notamsPrinted = true;
36
37    }
38  }
```

Výpis 2.10: Syntax funkcie volanej tlačidlom draw NOTAMs

2.6.2 Panel rightMidPanel



Obr. 2.6: Rozloženie komponentov v panely `rightMidPanel` uživateľského rozhrania

Tento panel obsahuje dva vnorené panely umiestnené pod sebou s komponentami na prácu s letovým plánom, pričom usporiadanie tohoto panelu je typu `BorderLayout`. Vrchný vnorený panel obsahuje komponenty typu `JLAbel` a `JTextField`. Spodný vnorený panel s usporiadaním komponentov typu `BoxLayout` obsahuje tlačidlá typu

`JButton` a jeden komponent typu `JComboBox<String>` ktorý zobrazuje zoznam súradníc letovej cesty.

Tlačidlo označené ako add point to Flight Path volá funkciu, ktorá pridáva súradnicu zadanú užívateľom do komponentov vrchného vnoreného panelu do letovej cesty objektu `flightPlan` pomocu jeho funkcie `addPathCoordinate(double latCoordinate, double lonCoordinate)` a tiež do zoznamu súradníc pomocou funkcie `addListToComboBox(List<String> input)`. Pred samotným pridaním súradnice prebieha kontrola či je súradnica zadaná v správnom tvare.

Draw Flight Plan je tlačidlo ktoré má na starosti zavolanie funkcie s logikou ktorá zabezpečuje vykreslenie užívateľom zadaného letového plánu do zobrazenej Mapy. Funkcia najprv skontroluje či letová cesta globálnej premennej `flightPlan` obsahuje viac ako jednu súradnicu. Ak áno, funkcia pokračuje ďalej vo výpočte. Funkcia zmaže aktuálny vykreslený letový plán a pomocou statickej funkcie `computeFlightCorridor(List<Double> flightPath)` triedy `AreaComputer` sa zo zadaných bodov letovej cesty vyráta nový letový koridor. Následne funkcia prejde dvoma za sebou idúcími for cyklami najprv súradnice letového koridoru a následne súradnice letovej cesty globálnej premennej `flightPlan`, pričom ich do zobrazenej mapy vykreslí pomocou funkcií objektu typu `Browser`. Letová cesta je vykreslená čiernou čiarou a okolo nej je vykreslený letový koridor farbou modrou tak ako je zobrazené na obrázku 2.4. Po vykreslení letového plánu sa for cyklom prejde premenná `notams` ktorá obsahuje objekty triedy `Notam` a pre každý objekt sa vyráta jeho premenná `crossingFlightCorridor` indikujúca kolíziu s letovým plánom. Týmto krokom je zaručené že sa pri zmene letového plánu znovu preráta jeho kolízia s oblasťami správ NOTAM a jednotlivé oblasti budú vykreslené správnou farbou.

Tlačidlo delete Flight Plan volá funkciu `deleteFlightPlan(ActionEvent e)` ktorá zmaže práve vykreslený letový plán. Funkcia premaže súradnice letovej cesty globálnej premennej `flightPlan` a taktiež zmaže súradnice ktoré sú uvedené v zozname súradníc letovej cesty v komponente typu `JComboBox<String>`. Na konci funkcia ešte for cyklom prejde jednotlivé objekty `Notam` uložené v premennej `notams` a každému sa nastaví premenná `crossingFlightCorridor` na hodnotu false a farba zobrazenia jeho oblasti v mape na zelenú.

Pod tlačidlom delete Flight Plan sa nachádza komponent typu `JComboBox<String>` zobrazujúci aktuálne nastevené súradnice letovej cesty. Slúži na prehľad užívateľa o súradniciach aktuálnej zadanej letovej cesty ale taktiež na zmazanie určitej vybranej súradnice pomocou tlačidla delete selected point. Súradnice sú v zozname zapísané v tvare „poradové číslo.lat: hodnota lon: hodnota“ pričom hodnota je zaokrúhlená vždy na tri desatinné miesta.

Tlačidlo delete selected point volá funkciu, ktorá najprv získa aktuálny označený záznam o súradnici letovej cesty a následne z neho získa poradové číslo súradnice

v letovej ceste. Na základe poradového čísla je súradnica vymazaná zo zoznamu súradníc typu `JComboBox<String>` ale taktiež zo zoznamu súradníc letovej cesty golbálnej premennej `flightPlan`. Syntax tejto funkcie je na výpisku 2.11.

```
1  private void deletePointFromPath(ActionEvent e){
2      String point = String.valueOf(pathPoints.getSelectedItem());
3      if(point.equals("null")){
4        log("Select point to remove");
5        return;
6      }
7      int pointNumber = Character.getNumericValue(point.charAt(0));
8      pathPointsList.remove(pointNumber - 1);
9      flightPlan.removePathCoordinate(pointNumber);
10
11     addListToComboBox(pathPointsList);
12     log("Point " +point.substring(6) + " removed from flightPath");
13 }
```

Výpis 2.11: Syntax funkcie volanej tlačidlom delete selected point

2.6.3 Panel rightbottomPanel

Obr. 2.7: Rozloženie komponentov v panely `rightBottomPanel` uživateľského rozhrania

Spodný z troch hlavných panelov má za úlohu prídavnú funkcionalitu a taktiež rieši komunikáciu programu s užívateľom. Skladá sa z dvoch vnorených panelov ktoré su uložené vertikálne pod sebou. Vrchný vnorený panel obsahuje jednu komponentu typu `JTextArea` ktorá zabezpečuje prídavnú komunikáciu programu s užívateľom pomocou funkcie `log(String log)`. Tejto funkcií sa vstupným argumentom zadá text ktorý funkcia následne do komponenty vypíše. Jedná sa najmä o informácie popisujúce správanie programu ako napríklad úspešné pridanie súradnice, chyba v otvorení súboru, súradnica zadaná v nesprávnom tvare a pod. Spodný vnorený panel obsahuje dve tlačidlá zoom out a zoom in. Tieto tlačidlá po kliknutí na ne zavolajú

funkciu `executeJavaScript(String JavaScriptCommand)` objektu `browser` typu `Browser` na vykonanie JavaScript výrazov. JavaScript výrazy ktoré sa dajú funkcií ako argument slúžia buď na priblíženie alebo na oddialenie zobrazenej mapy. Syntax funkcie volanej kliknutím na tlačidlo zoom in je na výpisku 2.12.

```
1 private void zoomIN(ActionEvent e){
2
3     browser.executeJavaScript("zoom=map.getZoom()");
4     browser.executeJavaScript("map.setZoom(++zoom)");
5 }
```

Výpis 2.12: Syntax funkcie volanej tlačidlom zoom in

3 ZÁVER

Táto práca sa v teoretickej časti obsiahnutej v kapitole 1 zaoberá správou NOTAM využívanou v letectve. Je v nej rozobraná problematika neštrukturovanej textovej správy NOTAM v porovnaní s novšou digitálnou podobou značenou ako D–NOTAM. Ďalej sú v nej popísané jednotlivé špeciálne typy správ NOTAM a podrobne spracované významy jednotlivých položiek správy, z ktorých sa skladá. Na konci teoretickej časti je uvedená ukážková správa NOTAM a jej korektné dekódovanie.

V práci bol vytvorený program ktorý, je plne funkčný, písaný v programovacom jazyku Java. Program je schopný spracovať neštrukturované textové správy NOTAM do podoby vhodnej pre strojové spracovanie, vykresliť oblasti platnosti jednotlivých správ do mapy a detekovať kolíziu týchto oblastí s letovým plánom. Letový plán si môže užívateľ zadefinovať v užívateľskom prostredí. Správy NOTAM program načítava zo súboru ktorý taktiež definuje užívateľ. Pri tvorbe programu bol kladený dôraz najmä na spracovanie zložitých oblastí, ktoré sú popísane časťami kruhov alebo úsekmi idúcimi pozdĺž hraníc štátov. Program je schopný spracovať správy NOTAM rozdelené do viacerých častí, ako aj tie popisujúce viac ako jednu oblasť. Oblasti správ sú do mapy vykreslené zelenou farbou. V prípade, že letový plán prechádza cez oblasť správy, táto oblasť sa vykreslí červenou farbou, ktorá indikuje kolíziu a nebezpečenstvo. Program je vyexportovaný do spustiteľného JAR súboru, ktorý obsahuje všetky potrebné súbory pre správne fungovanie. Tento súbor je spolu so súbormi jednotlivých tried obsahujúcich ich zdrojové kódy a ďalšími súbormi využitými pri tvorbe programu k dispozícií v prílohe A. Jednotlivé triedy a metódy programu sú spolu s ukážkami rozobrané v kapitole 2 zaoberajúcej sa programom. Testovanie a ladenie bolo vykonané na približne sto správach poskytnutých vedúcim práce.

Motiváciou na vypracovanie tejto práce a tvorbu programu bolo jeho reálne využitie v letectve. V krajinách, ktoré v dnešnej dobe stále využívajú neštrukturovanú textovú podobu správy NOTAM je povinnosťou pilota si pred letom naštudovať množstvo papierov obsahujúcich správy NOTAM, čo z časového hľadiska často nie je možné a tým pádom dochádza k strate informácií významných práve pre bezpečné prevedenie letu. Program vytvorený v rámci práce je schopný prebrať túto povinnosť pilota a správy NOTAM vykresliť v digitálnej grafickej podobe do mapy, ktorá bude pilotovi k dispozícii. Taktiež dokáže vykresliť letový plán a vyznačiť oblasti predstavujúce potencionálne nebezpečenstvo pre let. Hlavným prínosom práce a programu vytvoreného v rámci práce je teda zamedzenie straty informácií spôsobenej ľudským faktorom pri spracovaní správ NOTAM, ale aj zefektívnenie a zjednodušenie predletovej prípravy a samotného riadenia letu.

LITERATÚRA

[1] Wikipédia a prispievateľia. *Application programming interface* [online]. 5. 11. 2016 [cit. 23. 10. 2016]. Dostupné z URL: <https://en.wikipedia.org/wiki/Application_programming_interface>.

[2] AERONAUTICAL INFORMATION SERVICES–AERONAUTICAL INFORMATION MANAGEMENT STUDY GROUP. *FIFTH MEETING–NOTAM GUIDANCE* [online]. 31. 10. 2011 [cit. 23. 10. 2016]. Dostupné z URL: <http://www.icao.int/safety/ais-aimsg/aisaim%20meeting%20metadata/ais-aimsg%205/sn%204%20complete%20rev.pdf>.

[3] AERONAUTICAL INFORMATION SERVICES–AERONAUTICAL INFORMATION MANAGEMENT STUDY GROUP. *FIRST MEETING–DIGITAL NOTAM* [online]. 24. 11. 2008 [cit. 23. 10. 2016]. Dostupné z URL: <https://www.google.cz/url?sa=t&rct=j&q=&esrc=s&source=web&cd=8&ved=0ahUKEwiw-4O_jYXQAhWDBZoKHWSPBogQFghXMAc&url=http%3A%2F%2Fwww.icao.int%2Fsafety%2Fais-aimsg%2FMeetings%2FIPs%2FAIS-AIMSG.1.IP.004.en.doc&usg=AFQjCNHiOJudUNoVWOZD7EV6b-oLx00x6Q&sig2=Iq43g24l0n8gJtY2113zpw&cad=rja>.

[4] AIR TRAFFIC MANAGEMENT SUB GROUP OF APANPIRG(ICAO). *THIRD MEETING–NOTAM PROLIFERATION* [online] 20-24. 5. 2013 [cit. 23. 10. 2016]. Dostupné z URL: <http://www2010.icao.int/APAC/Meetings/2013_atmsg1/AI6%20IP12%20NOTAM%20Proliferation.pdf>.

[5] EUROCONTROL. *Digital NOTAM* [online]. 2010 [cit. 23. 10. 2016]. Dostupné z URL: <http://www.skybrary.aero/bookshelf/books/2073.pdf>.

[6] EUROCONTROL. *FIR/UIR in the Lower Airspace (EUROCONTROL Member States)* [online]. 7. 1. 2016 [cit. 23. 10. 2016]. Dostupné z URL: <https://www.eurocontrol.int/sites/default/files/content/documents/nm/cartography/07012016-firuir-lower-airspace-ectl.pdf>.

[7] EUROCONTROL AND FEDERAL AVIATION ADMINISTRATION. *Digital NOTAM* [online]. 2013 [cit. 23. 10. 2016]. Dostupné z URL: <http://www.aixm.aero/page/digital-notam>.

[8] EUROCONTROL. *SNOWTAM Harmonisation Guidelines* [online]. 24. 9. 2010 [cit. 23. 10. 2016]. Dostupné z URL: <http://www.skybrary.aero/bookshelf/books/2074.pdf>.

[9] FÁBRY, Ľubomír, Matej ANTOŠKO a Peter KORBA. *Procedurálne riadenie letovej prevádzky.* Vydanie prvé. Edícia vedeckej a odbornej literatúry (Sdružení požárního a bezpečnostního inženýrství), 164 s., 2015. ISBN 978-80-7385-160-6.

[10] INTERNATIONAL CIVIL AVIATION ORGANIZATION. *Aeronautical Information Services Manual, Doc 8126 AN/872*[online]. 2003 [cit. 23. 10. 2016]. Dostupné z URL: <http://www2010.icao.int/NACC/Documents/Meetings/2014/ECARAIM/REF09-Doc8126.pdf>.

[11] LONDON HEATHROW AIRPORT *Heathrow Traffic Statistics*[online]. 1.1.2005–1.9.2016 [cit. 23. 10. 2016]. Dostupné z URL: <http://www.heathrow.com/file_source/Company/Static/Excel/traffic_statistics-monthly-heathrow-200501_to_201609.xlsx>.

[12] MINISTERSTVO DOPRAVY ČESKÉ REPUBLIKY–Úřad pro civilní letectví. *LETECKÝ PŘEDPIS L15 O LETECKÉ INFORMAČNÍ SLUŽBĚ*[online]. Posledná zmena č. 5/ČR, 1. 5. 2014. [cit. 23. 10. 2016]. Dostupné z URL: <http://lis.rlp.cz/predpisy/predpisy/dokumenty/L/L-15/data/print/L-15_cely.pdf>.

[13] MINISTERSTVO DOPRAVY ČESKÉ REPUBLIKY–Úřad pro civilní letectví. *LETECKÝ PŘEDPIS L7030 - EVROPSKÉ (EUR) REGIONÁLNÍ DOPLŇKOVÉ POSTUPY*[online]. Posledná zmena č. 6/ČR, 17. 10. 2013. [cit. 23. 10. 2016]. Dostupné z URL: <http://lis.rlp.cz/predpisy/predpisy/dokumenty/L/L7030/data/print/L7030_cely.pdf>.

[14] SCHILDT, Herbert. *Mistrovství–Java.* Brno:Computer Press, 2014. ISBN 978-80-251-4145-8.

[15] SKYBRARY. *BIRDTAM*[online]. [cit. 23. 10. 2016]. Dostupné z URL: <http://www.skybrary.aero/index.php/BIRDTAM>.

[16] U.S. DEPARTEMENT OF TRANSPORTATION–FEDERAL AVIATION ADMINISTRATION. *Notices to Airmen (NOTAM)*[online]. 18. 12. 2015. [cit. 23. 10. 2016]. Dostupné z URL: <http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=6D5A3913FE2E81082FF1F570F42E2C85?doi=10.1.1.728.6781&rep=rep1&type=pdf>.

[17] ZITKO, Karel a Milan VACÍK. *Učebnice létání: příručka pro výcvik soukromého pilota letounů–PPL(A).* 5., rozš. vyd. Praha: Vintage Aviation, 132 s., 2014. ISBN 978-80-260-6640-8.

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

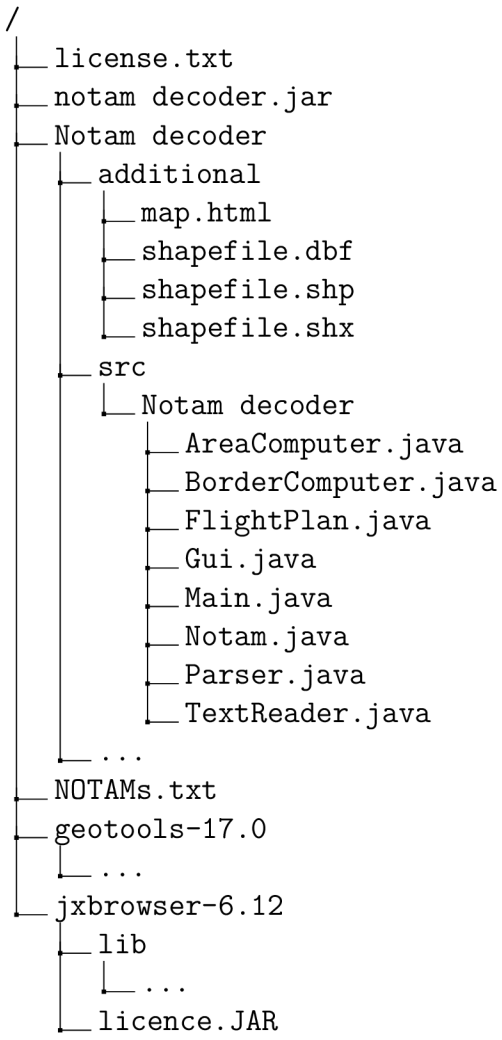
AFTN	Aeronautical Fixed Telecommunication Network
AIS	Aeronautical Information Service
AIXM	Aeronautical Information Exchange Model
API	Application programming interface
D–NOTAM	Digital NOTAM
EAD	European AIS Database
FIR	Flight Information Region
FIS	Flight information service
FL	Flight Level
ft	foot
GNSS	Global Navigation Satellite System
ICAO	International Civil Aviation Organization
IFR	Instrument Flight Rules
JAR	Java Archive
m	meter
NATO	North Atlantic Treaty Organization
NM	Nautical Mile
NOTAM	Notice To Airmen
NOTAMC	Notice To Airmen Cancellation
NOTAMN	Notice To Airmen New
NOTAMR	Notice To Airmen Replacement
PIB	Pre-flight Information Bulletins
UIR	Upper Information Region
UTC	Coordinated Universal Time
VFR	Visual Flight Rules

ZOZNAM PRÍLOH

A Obsah priloženého CD **54**

A OBSAH PRILOŽENÉHO CD

Všetky prílohy k práci sú dodané v podobe súborov nahratých na CD. Adresárová štruktúra tohto CD je zobrazená na obrázku A.1.



Obr. A.1: Adresárová štruktúra priloženého CD

Súbor „license.txt“ obsahuje licenciu k programu.

Súbor s názvom „notam decoder.jar“ predstavuje spustiteľný JAR súbor, ktorý obsahuje všetky potrebné súbory na správne fungovanie programu a teda je jeho spustenie nezávislé od prídavných súborov.

V adresári „Notam decoder“ sa nachádzajú všetky súbory potrebné k spusteniu a zobrazeniu programu vo vývojovom prostredí ako JAVA projekt. Obsahuje adresár „src“, v ktorom sa nachádzajú zdrojové kódy všetkých ôsmich tried, ktorých logika je rozobraná v kapitole 2. Taktiež sa tu nachádza aj zdrojový kód triedy pomenovanej ako „Main.java“, ktorá obsahuje metódu `main(String[] args)` na

spustenie programu. Adresár „Notam decoder“ obsahuje tiež adresár „additional“, ktorý obsahuje súbor „map.html“, ktorý je zobrazený v užívateľskom rozhraní ako mapa. Okrem neho sa v adresári nachádzajú súbory potrebné na vykreslenie oblastí idúcich pozdĺž hraníc štátov.

Súbor „NOTAMs.txt“ obsahuje tri správy NOTAM pre ukážku funkčnosti programu. Tieto správy sú syntetické–neobsahujú údaje, ktoré boli reálne použité v správach NOTAM využitých v letovej prevádzke.

Zvyšné dva adresáre obsahujú externé knižnice, ktoré sú potrebné pre správnu funkčnosť programu.